Haute Ecole de Gestion & Tourisme
Hochschule für Wirtschaft & Tourismus

Filière Informatique de gestion
Studiengang Wirtschaftsinformatik

Travail de Bachelor 2014

Mapper l'effort physique grâce au Hexoskin

Étudiant : Vincent / Pont

Professeur : Michael / Schumacher

Déposé le : 28 juillet 2014

www.hes-so.ch

Résumé

Des chaussures qui comptent nos pas, des montres et bracelets intelligents qui nous assistent durant notre séance de sport, les évolutions technologiques du 21ème siècle sont tangibles et inépuisables. Elles témoignent d'une avidité constante de modernité.

Ainsi, le nouveau vêtement Hexoskin est muni de différents capteurs comme : ECG, rythme cardiaque, volume respiratoire, podomètre, etc. Il enregistre par le biais d'un appareil toutes les données retransmises de ses capteurs. Ces éléments sont ensuite transférés sur l'ordinateur pour être analysés.

De ce fait, il est désormais possible d'examiner une séance de sport sous toutes ses formes avec beaucoup de détails afin de savoir si celle-ci a été performante.

Mon travail de Bachelor consista à mapper l'effort physique sur un plan cartographié par le biais du système GPS embarqué d'un smartphone et d'analyser les données issues des capteurs. Il fallut donner une large représentation de toutes ces données au sportif pour qu'il puisse correctement analyser les paramètres physiques de son entraînement.

Remerciements

Je tenais à remercier différents acteurs qui ont su m'orienter et m'aider pour le développement de ce travail de Bachelor, notamment :

Monsieur Michael Schumacher et Monsieur Bruno Alves qui m'ont épaulés pour le respect des objectifs et qui m'ont guidés tout au long de ce travail.

Anthony Bernardo qui a contribué à la réalisation d'un problème du développement sur le Cloud de Google.

Christophe Hadorn pour son expérience sur PhoneGap et sa nécessaire contribution.

Enfin, la communauté Stackoverflow qui a su répondre à mes différentes questions.

Glossaire

Hexoskin

Marque déposée du vêtement intelligent que j'ai utilisé pour mon travail de Bachelor.

Repository

Emplacement où l'on stocke généralement des fichiers ou différentes données informatiques.

Cloud

Nuage en français, représentant des ressources informatiques rendues publiques ou privées.

Datastore

Cloud de Google, il est en NOSQL.

Appengine

Est une plateforme de conception et d'hébergement d'applications.

Api

Interface de programmation qui est un ensemble de fonctions, procédures ou classes, mise à disposition des programmes informatiques par une bibliothèque logicielle[1].

Maven

Créer par Apache[2], il est un outil libre pour la gestion et l'automatisation de production des projets ou logiciels Java.

Endpoint

Sorte d'interface qui permet de relier l'ensemble des applications (web, mobiles…) en générant une Api[3].

[1] Source : http://icp.ge.ch/sem/cms-spip/spip.php?article956
[2] Glossaire : Apache

Android

Est un système d'exploitation pour smartphone.

Product owner

Le Product Owner est le porteur du produit. Il possède une vision complète de celui-ci[4].

Product backlog

C'est un document qui liste les besoins et exigences du client.

Java

Est un langage de programmation.

Apache

Est une organisation à but non lucratif qui développe des logiciels open source[5].

Rest

Representational State Transfer en anglais, est un style d'architecture pour les systèmes hypermédia distribués[6].

Objectify

Librairie qui aide à travailler avec le Datastore. N'est pas géré par Google.

Browser

Est un navigateur web (Chrome, Mozilla, Internet Explorer).

HTML

Est le format de données conçu pour représenter des pages web.

[3] Source : http://fr.openclassrooms.com/informatique/cours/montez-votre-site-dans-le-cloud-avec-google-app-engine/cloud-endpoints-creez-des-apis-et-generez-des-apps-android-iphone
[4] Source : http://freddy.yimo.fr/scrum-le-vocabulaire-tournant-autour-du-scrum/
[5] Source : http://fr.wikipedia.org/wiki/Apache_Software_Foundation
[6] Source : http://fr.wikipedia.org/wiki/Representational_State_Transfer

CSS

Les feuilles de styles (en anglais "*Cascading Style Sheets*", abrégé CSS) sont un langage qui permet de gérer la présentation d'une page Web[7].

Framework

Est un ensemble de logiciel ou composants qui aident au développement. Il est également un espace de travail modulaire.

Appstore

Est une plateforme de téléchargement pour mobiles.

W3C

Est une organisation non lucrative, fondée en 1994, qui définit des recommandations pour les technologies du web[8].

Pluggin

Est un module d'extension.

Login/Logout

Décrit en anglais une façon qu'un utilisateur à pour se connecter/déconnecter d'un site ou d'une application.

USB

Universal Serial Bus est une norme relative à un bus informatique en transmission série qui sert à connecter des périphériques informatiques à un ordinateur[9].

Bug

Terme pour désigner un défaut de conception.

[7] Source : http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/internet-css-4050/
[8] Source : http://glossaire.infowebmaster.fr/w3c/
[9] Source : http://fr.wikipedia.org/wiki/USB

Listener

Ecouteur en français il est utilisé en programmation quand un changement d'états est intercepté de la part d'un objet.

Bluetooth

Bluetooth est une spécification de l'industrie des télécommunications. Elle utilise une technique radio courte distance destinée à simplifier les connexions entre les appareils électroniques[10].

Json

Json est un format de données textuelles, générique, dérivé de la notation des objets du langage ECMAScript[11].

JQuery

jQuery est une bibliothèque JavaScript libre qui porte sur l'interaction entre JavaScript et HTML et a pour but de simplifier des commandes communes de JavaScript[12].

Phonegap

Est un framework open-source de développement mobile développé par Nitobi Software.

Ajax

Il désigne un nouveau type de conception de pages Web permettant l'actualisation de certaines données d'une page sans procéder au rechargement total de cette page[13].

Jsp

est une technique basée sur Java qui permet aux développeurs de créer dynamiquement du code HTML, XML ou tout autre type de page web[14].

[10] Source : http://fr.wikipedia.org/wiki/Bluetooth
[11] Source : http://fr.wikipedia.org/wiki/JSON
[12] Source : http://fr.wikipedia.org/wiki/JQuery
[13] Source : http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/high-tech-ajax-3998/

Php

Est un langage de programmation compilé à la volée libre principalement utilisé pour produire des pages Web dynamiques via un serveur http[15].

NoSQL

NoSQL signifie "Not Only SQL", littéralement "pas seulement SQL". Ce terme désigne l'ensemble des bases de données qui s'opposent à la notion relationnelle des SGBDR[16].

Table des matières

[14] Source : http://fr.wikipedia.org/wiki/JavaServer_Pages
[15] Source : http://fr.wikipedia.org/wiki/PHP
[16] Source http://blog.neoxia.com/nosql-5-minutes-pour-comprendre/

Table des figures

1 Introduction

1.1 Contexte

Après trois année d'études à la HES-SO Valais, il nous est demandé de mettre en pratique toutes nos connaissances dûment acquises pour notre travail de Bachelor.

Durant le dernier semestre divers thèmes nous sont proposés et il donnait la possibilité de choisir parmi ceux-ci.

Ainsi, un des thèmes qui a retenu mon intérêt fut « Mapper l'effort physique grâce au Hexoskin[17] ». Il concernait de nouvelles technologies dans le domaine du sport que je pratique assez régulièrement.

1.2 Objectif du projet

L'objectif de ce travail de Bachelor est de calculer et d'afficher l'effort physique sur un plan cartographié dans le domaine du sport d'endurance.

Il a fallu donc analyser les différentes données des capteurs et les présenter correctement sous forme de graphiques pour faciliter son analyse. Toutes les données devront être sauvegardées dans Google Cloud[18].

1.3 Délivrables

Deux applications ont été demandées dans le cahier des charges soit :

Une application mobile sur Android[19] qui enregistrera les informations de la séance de sport comme : la durée, la distance parcourue, les calories brûlées, la vitesse. Elle affichera et enregistrera également les données de localisation sur une carte et dessinera le trajet parcouru sur celle-ci en temps réel.

[17] Glossaire : Hexoskin
[18] Glossaire : Cloud
[19] Glossaire : Android

Une application web qui, elle, affichera les données des capteurs Hexoskin et couplera ces données avec celles d'Android pour donner des informations complètes et pertinentes au sportif. L'utilisateur pourra également comparer sa séance de sport et revoir le trajet parcouru sur une carte avec les différentes données inscrites directement sur le trajet.

1.4 Acteurs

Étudiant	Pont Vincent
Professeur responsable	Michael Schumacher
Proposition du thème	Bruno Alves

1.5 Cahier des charges

Le cahier des charges décrit toutes les étapes à réaliser. Il a été accepté à l'unanimité par toutes les parties.

2 Recherches et Analyses

2.1 Hexoskin

Avant de me lancer directement dans la phase de développement, je dus comprendre le fonctionnement de ce vêtement pour pouvoir y intégrer ma propre solution.

2.1.1 Fonctionnement

Le vêtement est muni de différents capteurs ainsi que d'un appareil qui enregistre toutes les données retransmises par ceux-ci. Cet appareil doit être au préalable rechargé via USB[20] pour pouvoir fonctionner.

L'étape suivante est d'installer un programme sur son ordinateur qui fera la synchronisation des données depuis notre ordinateur jusqu'au site https://my.hexoskin.com/. Une fois l'installation finie nous pouvons télécharger l'application mobile faite par Hexoskin disponible sur IPhone ou Android pour lier notre appareil Hexoskin via Bluetooth[21].

[20] Glossaire : USB
[21] Glossaire : Bluethoot

Il ne nous reste plus qu'à mettre notre gilet, activer le Bluetooth et le lier à l'appareil Hexoskin pour avoir directement un aperçu sur notre mobile des données qui sont enregistrées.

La dernière étape consiste à brancher notre appareil à notre ordinateur et à attendre que la synchronisation s'opère pour pouvoir ensuite analyser notre séance sur le site https://my.hexoskin.com/.

2.1.2 L'api Hexoskin

Après compréhension de l'architecture, je choisis d'intégrer mon application Android au même endroit que l'application Hexoskin via Bluetooth. Malheureusement ce ne fut possible qu'après demande auprès des techniciens Hexoskin.

A partir de là, je me suis concentré sur leur Api[22]. Cette Api fonctionne en Rest[23]. Il suffisait donc de faire des requêtes au moyen d'un compte et d'une clé de développeur pour recevoir les données sous format Json[24] de mes capteurs.

Pourtant, le fait de n'avoir pu passer directement par le Bluetooth créa certains problèmes de synchronisation des données que j'expliquerai ultérieurement dans mon travail.

2.2 Applications natives ou web

Il m'a été demandé de réaliser une analyse sur les différentes manières de concevoir une application mobile, d'en tirer les avantages et inconvénients de chacune d'elles et au final d'en sélectionner une.

2.2.1 L'analyse

Pour mon travail de Bachelor, je devais réaliser une application sur mobile pour afficher et mapper l'effort physique. En conséquence, pour atteindre, une application qui jouerait le rôle de moniteur durant ma séance de sport et enregistrerait toutes les données de celle-ci.

[22] Glossaire : Api
[23] Glossaire : Rest
[24] Glossaire : Json

Dès lors, deux directions se sont offertes à moi pour réaliser cette application : emprunter le monde d'Android et de l'application native ou m'immerger dans le monde du langage web et utiliser un framework[25] pour déployer mon application sur les différentes plateformes mobiles (application web hybride).

Durant deux semaines, j'ai dû faire des recherches et analyser ces deux technologies et langages différents afin de choisir la meilleure proposition pour mon travail de Bachelor. Je devais également penser aux objectifs de ce projet et réfléchir à leur faisabilité d'après ce choix (Accès GPS sur le mobile, affichage Google Maps).

Source : http://www.journaldugeek.com/2013/01/17/applications-natives-contre-les-webapps-forces-et-faiblesses/

2.2.2 L'application native en quelques mots

Une application native est une application développée uniquement dans un des systèmes d'exploitation utilisé par les smartphones (iOS, Android, etc.).

Les avantages :

- ✓ Expérience-utilisateur, meilleure et totalement intégrée à la plate-forme ;
- ✓ Vitesse accrue ;
- ✓ Accès aux fonctionnalités de l'appareil (calendrier, contact, GPS, etc...) ;
- ✓ Mode offline.

[25] Glossaire : Framework

Les inconvénients :

- Développement sur une seule plate-forme ;
- Onéreux si utilisation du multiplateforme.

Exemple concret : Géolocalisation

Voici comment l'on obtient les données de localisation sur Android:

```
LocationManager lm = (LocationManager)getSystemService(Context.LOCATION_SERVICE);
Location location = lm.getLastKnownLocation(LocationManager.GPS_PROVIDER);
double longitude = location.getLongitude();
double latitude = location.getLatitude();
```

```
private final LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        longitude = location.getLongitude();
        latitude = location.getLatitude();
    }
}

lm.requestLocationUpdates(LocationManager.GPS_PROVIDER, 2000, 10, locationListener);
```

Source : http://stackoverflow.com/questions/2227292/how-to-get-latitude-and-longitude-of-the-mobiledevice-in-android

Ensuite, j'utiliserai ces données pour les intégrer à Google Maps. J'obtiendrai l'affichage d'une carte et par la suite la possibilité de dessiner un trajet du parcours que l'individu aura réalisé durant son entraînement.

2.2.3 L'application web en quelques mots

Une application web mobile (ou web app) est une application utilisée via Internet et possédant des fonctionnalités spécifiques pour les appareils mobiles. Elle est accessible via le navigateur Web de l'appareil mobile (Safari sur l'iPhone par exemple) et n'a pas besoin d'être téléchargée et installée sur l'appareil pour fonctionner. Les langages de programmation sont donc axés sur le web (HTML[26]/CSS[27] et JavaScript).

Source : http://www.gettoplisting.com/images/img/native-hybrid-iphone-app.jpg

Les avantages :

- ✓ Multiplateforme (acceptée par tous les browsers[28]) ;
- ✓ Si manque de moyens, très pertinent car moins cher à développer.

Les inconvénients :

- ❖ Vitesse plus lente que natif ;
- ❖ Pas de lien avec les fonctionnalités de l'appareil ;
- ❖ Moins bonne expérience-utilisateur.

[26] Glossaire : HTML
[27] Glossaire : CSS
[28] Glossaire : Browser

2.2.4 L'application hybride en quelques mots

Une application hybride est une application pour mobiles qui combine des éléments HTML5 sous forme de web application mobile. Les éléments d'une application native permettent d'utiliser les fonctionnalités natives des smartphones et d'être distribuée en tant qu'application sur les plateformes d'applications (App Store[29], Android Market)[30].

Les avantages :

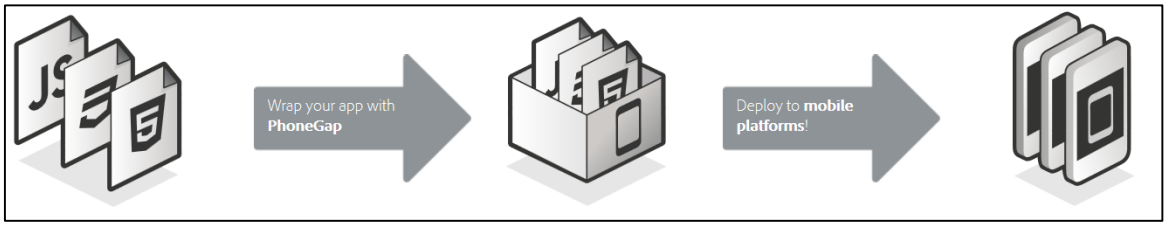
- ✓ Fonctionnalités du téléphone disponible à l'intermédiaire du framework ;
- ✓ Framework free et open source (PhoneGap[31]);
- ✓ Développement d'une application, mais déploiement sur plusieurs plateformes. Conséquence : gain de temps et d'argent.

Les inconvénients :

- ❖ Penser à l'affichage mobile lors du développement ;
- ❖ Apprentissage du framework (PhoneGap) ;
- ❖ Apprentissage JQuery[32] Mobile, Ajax[33] (Design mobile).

Les framework en quelques mots

Créer une application sur les différentes plateformes requière différents framework et langages de programmation. Le but d'un framework est d'aider le développeur à déployer leurs applications sur les différentes plateformes mobiles (Android, IPhone, Windows Phone, BlackBerry). Ils créent une sorte de pont entre l'application web et le téléphone mobile.



Source: http://phonegap.com/about/

[29] Glossaire : App Store
[30] Source : http://www.definitions-webmarketing.com/Definition-Application-hybride
[31] Glossaire : PhoneGap
[32] Glossaire : JQuery
[33] Glossaire : Ajax

Les différents framework

Durant mes recherches j'ai découvert différents framework dont le plus connu, PhoneGap. Ils fonctionnent de façon identique :

- PhoneGap ;
- Titanium Mobile ;
- Corona ;
- Sencha.

Analyse de PhoneGap

PhoneGap contient une large documentation de son Api. J'ai constaté des points essentiels comme le fait que PhoneGap arrive parfaitement à faire les liens entre les fonctionnalités du smartphone. Cet élément est primordial dans ma recherche pour récupérer des données du GPS.

Les composants de l'API :

Accéléromètre (mouvements de l'appareil), caméra, capture, contacts, notifications, géolocalisation, etc…

Dès lors, l'Api est assez idéal puisqu'elle fournit tout ce dont j'ai besoin pour mon projet.

Exemple concret : Géolocalisation

« La géolocalisation fournit des informations sur la localisation géographique du mobile, telles que la latitude et la longitude. Parmi les sources d'information de positionnement, on peut trouver le Global Positioning System (GPS), Cette API est basée sur la spécification W3C[34] Geolocation et Api Specification. »[35].

[34] Glossaire : W3C
[35] Source : http://docs.phonegap.com/fr/1.3.0/phonegap_geolocation_geolocation.md.html

Exemple rapide

```
// Fonction de callback onSuccess
//   Cette fonction reçoit en paramètre un objet `Position` qui contient
//   les coordonées GPS courantes
//
var onSuccess = function(position) {
    alert('Latitude : '              + position.coords.latitude          + '\n' +
          'Longitude : '             + position.coords.longitude         + '\n' +
          'Altitude : '              + position.coords.altitude          + '\n' +
          'Précision : '             + position.coords.accuracy          + '\n' +
          'Précision de l'altitude : ' + position.coords.altitudeAccuracy  + '\n' +
          'Direction : '             + position.coords.heading           + '\n' +
          'Vitesse : '               + position.coords.speed             + '\n' +
          'Date : '                  + new Date(position.timestamp)      + '\n');
};

// Fonction de callback onError, reçoit un objet PositionError
//
function onError(error) {
    alert('code : '    + error.code    + '\n' +
          'message : ' + error.message + '\n');
}

navigator.geolocation.getCurrentPosition(onSuccess, onError);
```

Source : http://docs.phonegap.com/fr/1.3.0/phonegap_file_file.md.html#File

Toutes les données sont retransmises par l'intermédiaire de PhoneGap (JavaScript[36]). Une fois ces données récupérées il me suffit de les gérer pour que, par la suite, elles s'intègrent dans l'API Google Maps (Création du plan, création d'un chemin sur le plan pour le sport d'endurance).

Je vais également utiliser un système de fichiers pour sauvegarder les données reçues du GPS. Cela pourrait être judicieux dans le cas où la connexion s'interrompt et évite de perdre les données précédemment reçues.

[36] Glossaire : JavaScript

Google Maps API :

Création de la carte en JavaScript :

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
    <style type="text/css">
      html { height: 100% }
      body { height: 100%; margin: 0; padding: 0 }
      #map-canvas { height: 100% }
    </style>
    <script type="text/javascript"
      src="https://maps.googleapis.com/maps/api/js?key={API_KEY}&sensor=SET_TO_TRUE_OR_FALSE">
    </script>
    <script type="text/javascript">
      function initialize() {
        var mapOptions = {
          center: new google.maps.LatLng(-34.397, 150.644),
          zoom: 8
        };
        var map = new google.maps.Map(document.getElementById("map-canvas"),
            mapOptions);
      }
      google.maps.event.addDomListener(window, 'load', initialize);
    </script>
  </head>
  <body>
    <div id="map-canvas"/>
  </body>
</html>
```

Source : https://developers.google.com/maps/documentation/javascript/shapes#polyline_add

Création d'un chemin/trajet avec l'aide de « Polylines » :

```
function initialize() {
  var mapOptions = {
    zoom: 3,
    center: new google.maps.LatLng(0, -180),
    mapTypeId: google.maps.MapTypeId.TERRAIN
  };

  var map = new google.maps.Map(document.getElementById('map-canvas'),
      mapOptions);

  var flightPlanCoordinates = [
    new google.maps.LatLng(37.772323, -122.214897),
    new google.maps.LatLng(21.291982, -157.821856),
    new google.maps.LatLng(-18.142599, 178.431),
    new google.maps.LatLng(-27.46758, 153.027892)
  ];
  var flightPath = new google.maps.Polyline({
    path: flightPlanCoordinates,
    geodesic: true,
    strokeColor: '#FF0000',
    strokeOpacity: 1.0,
    strokeWeight: 2
  });

  flightPath.setMap(map);
}

google.maps.event.addDomListener(window, 'load', initialize);
```

Source: https://developers.google.com/maps/documentation/javascript/shapes#polyline_add

PhoneGap avantages:

- Développement d'une seule application, mais déploiement sur plusieurs plateformes (Gain de temps et d'argent) ;
- Fonctionnalités de l'appareil disponibles ;
- Distribution de l'application sur les App stores. ;
- Gratuit et open source.

PhoneGap désavantages:

- PhoneGap ne supporte pas tous les fonctionnalités des appareils ;
- PhoneGap a toujours un retard par rapport aux applications natives quand il y a de nouvelles fonctionnalités ;
- L'application a le même design sur toutes les plateformes ;
- Leur design est générique et ne ressemble pas aux applications natives (expérience utilisateur).

Entretien avec Christophe sur PhoneGap

J'avais eu connaissance qu'un ancien collègue de classe avec utilisé PhoneGap pour l'un de ses travaux à Cyberlearn et j'ai décidé de lui demander conseils. Il m'a énuméré les principaux désavantages d'utiliser PhoneGap et les problèmes qu'il avait rencontrés.

Quelques inconvénients :

- Pas très pratique, tout est en ligne de commande ;
- Installation difficile ;
- JQueryMobile pour design (2-3 sem. d'apprentissage) ;
- Installation de pluggin[37] (Facebook) compliqué car installation sur les 3 environnements, donc travail triple ;
- Compilation dure et longue ;
- Avoir les environnements des plateformes pour développer.

[37] Glossaire : Pluggin

2.2.5 Comment choisir ?

Pour m'aider à choisir, je me suis posé les questions suivantes :

- Cette application doit-elle accéder à des fonctionnalités matérielles de l'appareil ?
- L'application a-t-elle besoin de beaucoup de ressources pour s'exécuter ?
- Pourquoi je veux faire du mobile ?
- A quelle population je m'adresse ?
- Le temps à disposition ?
- Le budget ?

Source : http://test_corp.beapp.fr/wp-content/uploads/2012/09/tumblr_m0dlz2ZM161r1n03z.jpeg

Certainement ces questions se posent lorsqu'on veut aller plus loin avec cette application ou lorsqu'on la développe pour une entreprise (mise à jour, déploiement App Store, vitesse, coûts et besoins graphiques).

J'ai donc fait un comparatif en tenant compte de mes besoins et de l'impact sur mon travail de Bachelor.

	Accès aux fonctionalités de l'appareil	*Temps d'apprentissage*	*Hors connexion*	*Vitesse app*	*Coût*	*Validation application*	*App Store*
Application native	Complet	Faible	Fonctionne	Très rapide	Cher	Obligatoire	Disponible
Web app Hybride	Complet (Geolocalisation, système de fichiers)	Moyen à long	Fonctionne	Vitesse native	Raisonnable	Obligatoire	Disponible
	Importance/Impacte TB	*Fort*					
		Moyen					
		Faible					

Figure 1: Comparatif application native et web app hybride

2.2.6 Conclusion

Cette analyse m'a permis de comprendre davantage les différentes manières de concevoir une application. J'ai pu distinguer les faiblesses et qualités de chacune pour m'aider dans ma prise de décision.

L'application native possèdera un seul framework (Android Studio) et comprendra moins de langages de programmation différents (Java[38]). Pour l'application web j'utiliserais plusieurs langages et framework différents (Ajax, JavaScript, JQueryMobile, PhoneGap) donc moins d'homogénéité et plus d'apprentissage.

Le fait de partir sur une application web hybride engendrerait un temps d'apprentissage supplémentaire. Je préfère donc utiliser ce temps pour faire une application native sur Android plus complète que de réaliser une application moins bonne, voire inachevée.

[38] Glossaire : Java

3 Application Android

3.1 Résumé

L'application Android servira comme moniteur durant la séance de sport. Elle enregistrera plusieurs données essentielles comme :

- Durée de la séance ;
- Distance parcourue ;
- Calories brûlées ;
- Vitesse ;
- Données de localisation.

L'application utilise le système embarqué de GPS du smartphone pour récupérer les données de localisation. Elle enregistrera tout au long de la séance les points de longitudes et latitudes, d'altitudes et de vitesses. C'est ainsi qu'avec ces données je pourrai mapper l'effort physique sur mon application web en reconstruisant le trajet effectué et également le fait de pouvoir comparer deux entraînements.

3.2 Écrans présentés

3.2.1 Login avec Google

C'est le premier écran présenté à l'utilisateur. C'est là qu'il peut se connecter avec son compte Google.

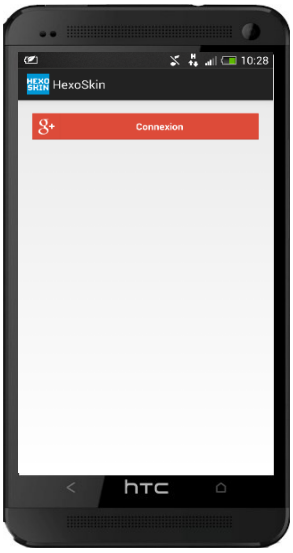


Figure 3: Écran login

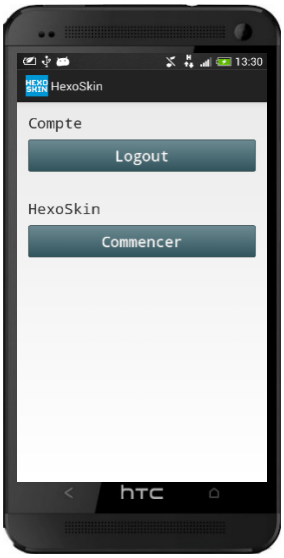


Figure 2 : Écran login suite

Lors d'une première connexion, il est demandé de choisir le compte de connexion pour utiliser l'application et autoriser celle-ci à recueillir des informations.

Par la suite, l'utilisateur pourra soit changer de compte en cliquant sur « Logout[39] », soit continuer sur l'application et passer à l'écran suivant.

3.2.2 Informations

Cet écran demande diverses informations à l'utilisateur par rapport à une liste de choix définie :

- Son sexe ;
- Son poids ;
- Son âge.

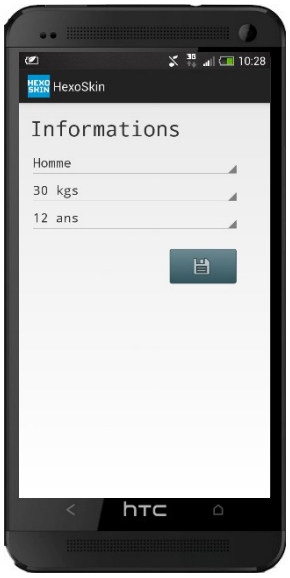


Figure 4: Écran informations

Ces données sont nécessaires aux calculs. Elles sont donc indispensables pour obtenir une précision optimale durant l'enregistrement des données. Le sportif n'a plus qu'à sauvegarder pour continuer à l'étape suivante.

3.2.3 Nouvelle séance

Tout d'abord, il est nécessaire d'activer la localisation/GPS dans les propriétés du smartphone. Si celle-ci n'est pas préalablement activée, un message signalera à l'utilisateur d'activer son GPS. Il ne pourra donc pas continuer si aucun GPS n'est mis en route.

[39] Glossaire : Login/Logout

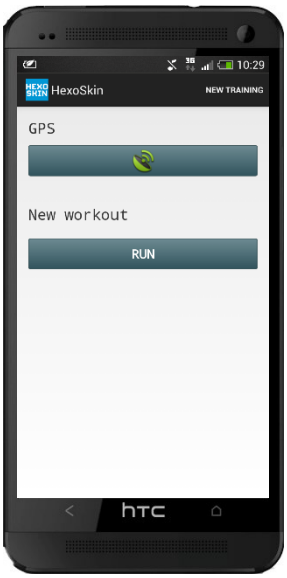


Figure 6: Écran nouvel entraînement

Figure 5: Écran paramètres localisation du smartphone

Pour une précision vraiment accrue, il est recommandé également d'activer la « Localisation et recherche de Google ». Après activation du GPS, l'utilisateur peut lancer sa séance de sport en cliquant sur « **RUN** ».

3.2.4 Carte et information séance

Cet écran sera présenté pendant l'entraînement. On y retrouvera la carte avec le chemin parcouru et les données calculées de la séance. Trois boutons sont présentés à l'utilisateur :

- **Play** : l'utilisateur démarre la séance. Le chronomètre se lance et les données GPS commencent à être enregistrées ainsi que les données calculées.
- **Pause** : l'utilisateur met en pause sa séance. Les données et le GPS sont mis en pause tant qu'il n'aura pas relancé en cliquant sur « Play ».
- **Stop** : l'utilisateur arrête sa séance de sport. Les données GPS et les calculs sont transférées vers l'écran « résumé séance ».

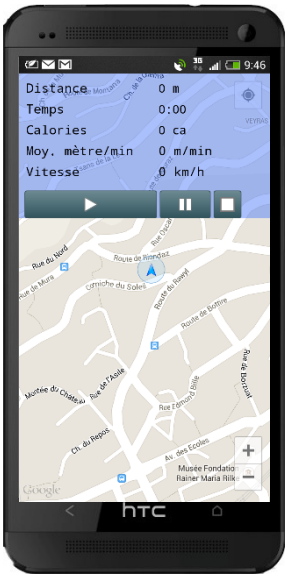


Figure 7: Écran carte et affichage données séance

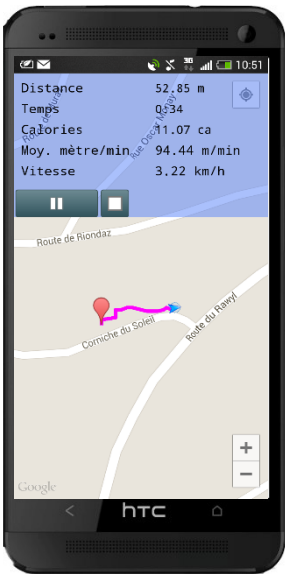


Figure 8: Écran carte, construction trajet

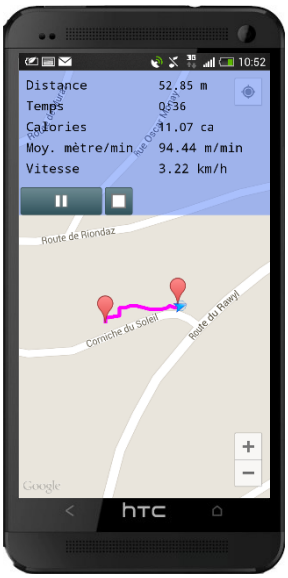


Figure 9: Écran carte, fin de la séance

On observe sur les figures le trajet du parcours qui se dessine au fur et à mesure que le coureur se déplace. Au début de la séance, un marqueur « Start » est placé pour savoir le lieu où la séance a débutée ainsi qu'un marqueur « End » pour voir où elle s'est terminée.

Pendant que j'ai développé mon application, j'ai remarqué qu'il fallait attendre un certain temps avant que le GPS ne se calibre parfaitement à l'endroit où je me trouvais. Cela avait pour conséquence de fausser certaines données comme la distance et les calories.

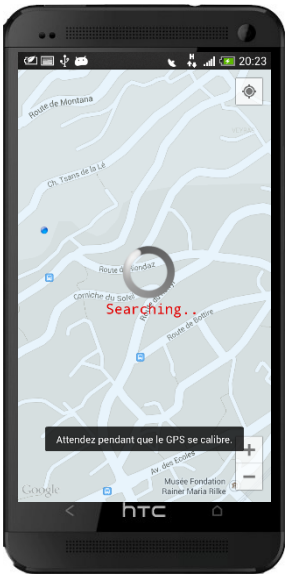


Figure 10: Écran avant carte, calibrage GPS

J'ai donc mis en place un temps d'attente indispensable avant que l'utilisateur ne puisse commencer sa séance. Il est donc informé que le GPS est en train de se calibrer « **Searching**.. » et il doit attendre environ 15 secondes. Dès que le calibrage est fini, l'utilisateur peut enfin démarrer sa séance de sport dans des conditions optimales.

Je me suis également aperçu que le GPS peut devenir capricieux et que le temps de calibrage que j'ai stipulé dans mon code n'était pas suffisant. Je me suis donc posé la question si ce problème venait de mon propre code ou si c'était vraiment le GPS en lui-même.

J'ai donc comparé mon écran et lancé l'application « Maps » de Google pour voir si j'étais positionné au même endroit. Ceci fut la preuve qu'il y avait un lien avec le GPS et non au code de mon application. Je devais être dans les environs de la commune de Veyras et le GPS me positionnait au centre de la ville de Sierre.

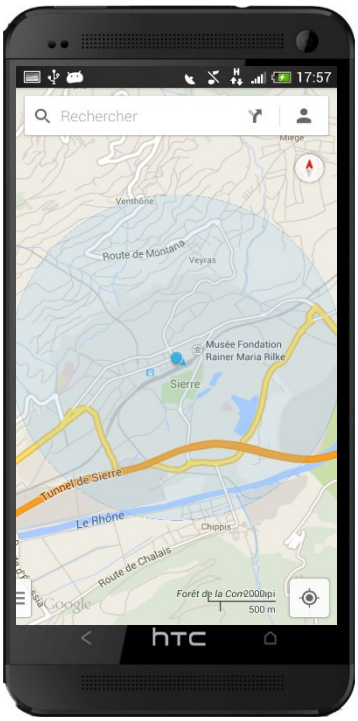


Figure 11: Application Maps de Google

Figure 12: Écran carte application Android

3.2.5 Résumé séance et sauvegarde dans cloud

Cet écran présente un résumé de la séance qui s'est déroulée. Une vitesse moyenne a été rajoutée car cette donnée est bien plus représentative à ce stade que la dernière vitesse enregistrée.

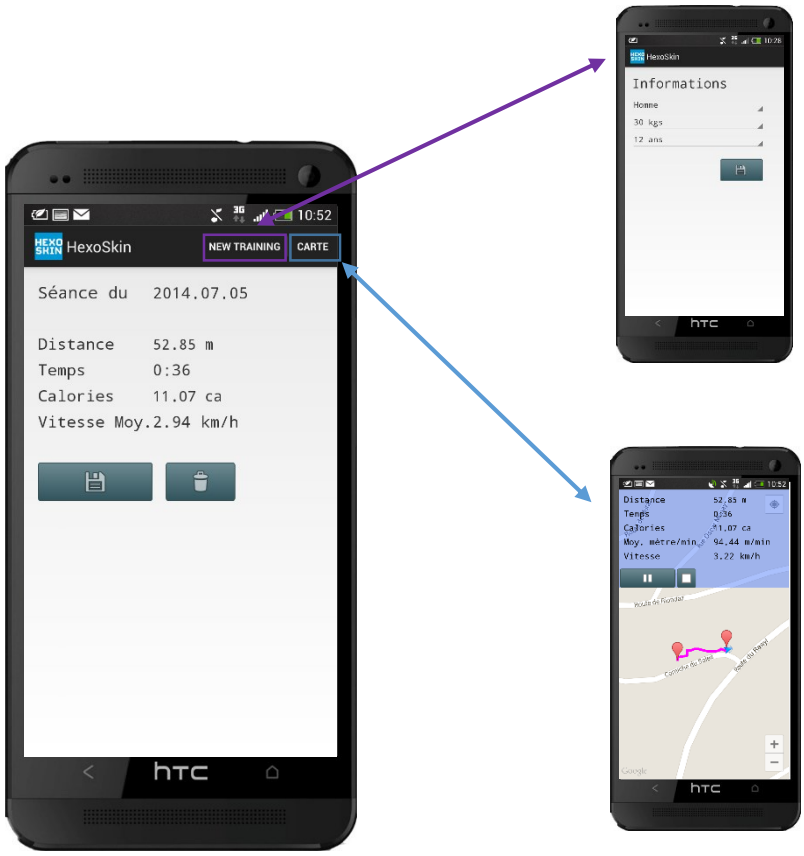


Figure 13: Écran résumé séance et sauvegarde dans le Cloud

Deux boutons sont présentés à l'utilisateur. Il peut sauvegarder cette séance dans le cloud ou la supprimer et dans ce cas elle ne sera pas enregistrée.

A noter que si la séance n'est pas sauvegardée, elle ne sera pas disponible sur l'application web.

De plus, l'utilisateur peut, depuis les menus en haut à gauche, relancer une nouvelle séance de sport ou revoir le trajet qu'il vient d'effectuer directement sur la carte. Le chemin parcouru sera bien évidemment sauvegardé lui aussi pour un visionnage plus précis sur l'application web.

3.3 Données présentées durant la séance

3.3.1 Durée séance

La durée de la séance est calculée par rapport à un chronomètre qui se lance dès que l'utilisateur clique sur le bouton « Play ».

Ce chronomètre servira également dans le calcul des calories brûlées et celui des mètres/min.

3.3.2 Distance séance

Ceci est la distance parcourue par la personne qui est en train de réaliser un effort physique. Elle est affichée en mètres et calculée par une méthode que j'ai utilisée depuis l'objet *Location*.

3.3.3 Calories brûlées

Les calories brûlées sont affichées par rapport à un calcul qui prend en compte plusieurs variables comme le sexe, le poids, le temps et ainsi que la moyenne des battements du cœur.

Pour la moyenne des battements du cœur, les données sont directement sur le vêtement. Elles ne sont pas disponibles en directe. Une moyenne des battements du cœur que pourrait faire une personne durant un effort physique est calculée.

3.3.4 Moyenne mètres/min

On calcule la distance parcourue selon un laps de temps et on fait une moyenne par minute. Cette information est disponible seulement durant la séance et n'est pas sauvegardée dans le Cloud puisque la vitesse en km/h est bien plus précise est pertinente pour l'utilisateur.

3.3.5 Vitesse

Au début, j'avais créé une méthode qui calculait la vitesse par rapport à une distance et à un temps. Un peu plus tard, j'ai trouvé qu'une méthode bien plus précise existait depuis l'objet *location* pour calculer la vitesse.

3.3.6 Google Maps

L'affichage de la carte est présenté à l'utilisateur par le biais du service Google Maps. Sur la carte, un trajet se dessinera automatiquement lors du déplacement du sportif. Ces données de localisation sont par la suite sauvegardées pour l'analyse et la reconstitution du trajet sur l'application web.

4 Application Web

4.1 Résumé

L'application web servira à coupler les données d'Android et celles des capteurs du vêtement Hexoskin.

Elle présentera à l'utilisateur des graphiques pertinents et la possibilité de comparer les entraînements passés pour en ressortir les points à améliorer. Il pourra également consulter la carte avec le trajet qu'il a parcouru avec des indices de performances directement affichés par-dessus.

Le site web sera programmé en Java EE/Jsp[40]. Ce choix a été décidé en raison que le Cloud de Google n'est pas encore assez optimisé pour du Php[41].

4.2 Pages présentées

4.2.1 Page login

Le login fonctionne comme pour l'application mobile. Pour se connecter à l'application web, l'utilisateur doit être en possession d'un compte Google. Après connexion avec son compte et une fois l'autorisation d'accès au site web obtenue, il sera dirigé automatiquement sur la page « entraînement ».

[40] Glossaire : Jsp
[41] Glossaire : Php

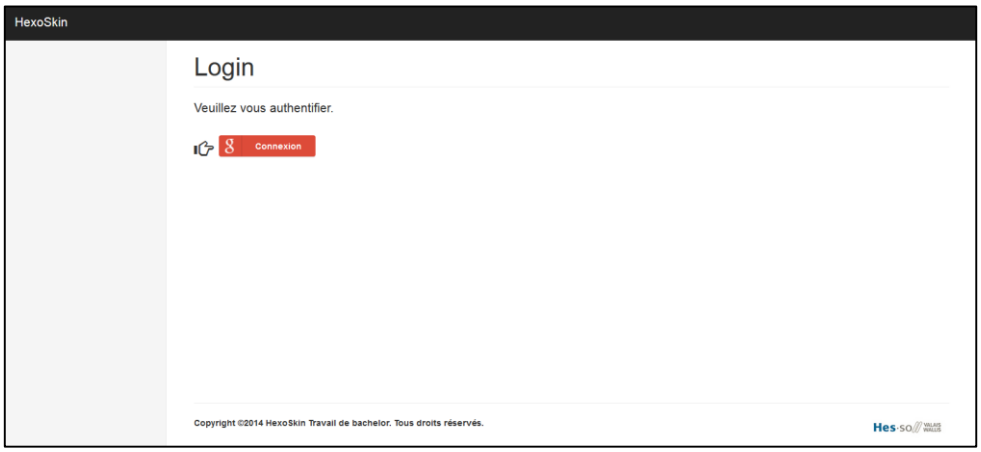


Figure 14: Page login site web

Il a également été implémenté une fonction de « Logout » pour que l'utilisateur puisse se déconnecter ou changer de compte. Après avoir cliqué sur « connexion », il est réorienté automatiquement sur la page login.

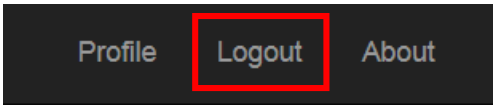


Figure 15: Menu site web, action Logout

4.2.2 Entraînement

C'est la première page présentée à l'utilisateur après qu'il se soit connecté avec son compte Google. Cette page présente à l'utilisateur l'entraînement le plus récent. Il aura une vue d'ensemble sur la séance.

Données de séance

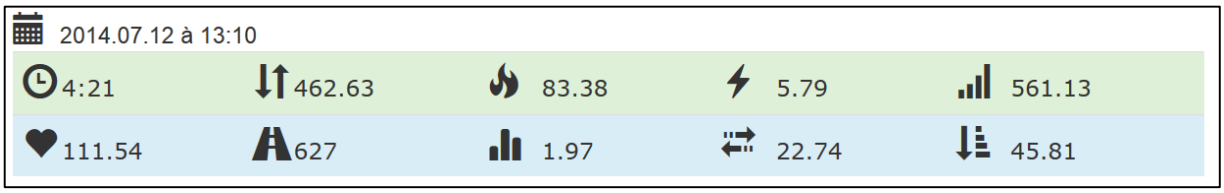


Figure 16: Page entraînement, données Android et Hexoskin

Deux types de données sont affichés. Les valeurs en vert représentent les données enregistrées par l'application Android et celles en bleues des capteurs du vêtement Hexoskin. On y retrouve :

- Le temps total réalisé ;
- La distance parcourue ;
- Les calories brûlées ;
- La vitesse moyenne en kilomètre/heure ;
- L'altitude moyenne en mètres ;

En bleu, nous avons les données propres au vêtement Hexoskin :

- La pulsation moyenne ;
- Les pas totaux ;
- Le volume tidal moyen en litres/inspiration ;
- La respiration moyenne par minute ;
- La ventilation moyenne en litres/minute.

L'utilisateur bénéficie déjà d'une bonne vue d'ensemble de toutes les valeurs moyennes enregistrées pendant son entraînement. Les moyennes ont été calculées manuellement pour représenter à l'utilisateur un meilleur jugement de la séance.

Carte

Une carte est affichée en dessous des informations de la séance. Elle y montre notamment le trajet parcouru lors du dernier entraînement.

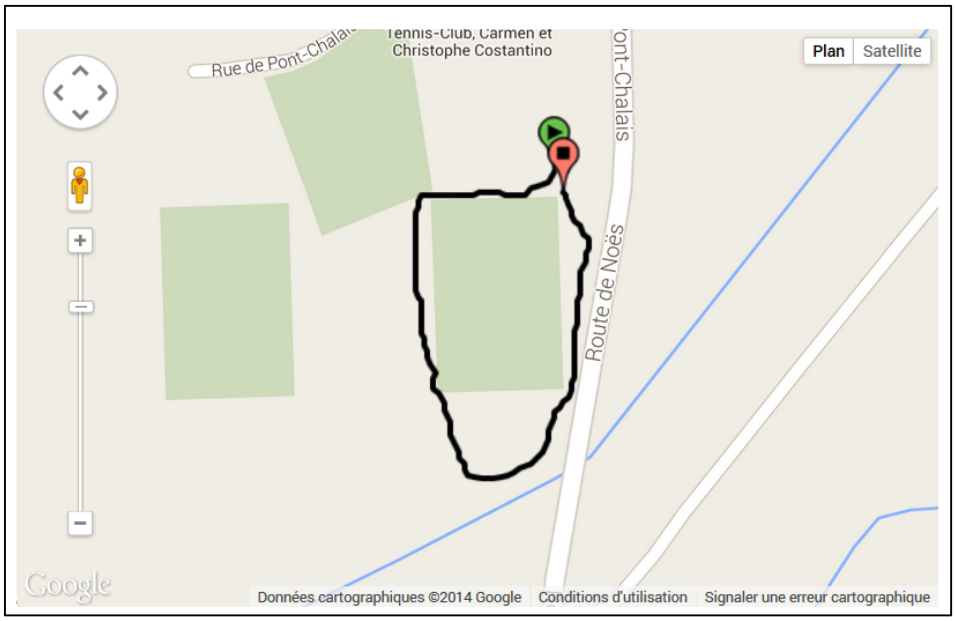


Figure 17: Page entraînement, présentation carte et trajet

On voit ci-dessus une représentation du parcours que j'ai réalisé à Pont-Chalais sur la commune de Sierre. On aperçoit également là où la séance a débuté et là où elle s'est terminée.

Les filtres

Une interactivité a été ajoutée pour que l'utilisateur puisse afficher différents éléments directement sur la carte à chaque point de localisation enregistré.

Pendant mes recherches sur Google Maps Api, j'ai analysé tous les objets mis à disposition par Google pour afficher et représenter des éléments sur une carte. Les options offertes étaient assez larges mais encore fallait-il que je puisse les utiliser dans le domaine du sport et représenter l'effort sur un trajet.

J'ai choisi en majeure partie la fonctionnalité « marqueur ». Cette option permet de rajouter à un point de location donné une icône représentative avec la possibilité de rajouter une info-bulle liée à celui-ci. Cette fonctionnalité me permettait donc, par différentes couleurs de marqueur, de représenter une vitesse faible ou forte directement sur le trajet. Le sportif peut également connaître sa vitesse précise en cliquant directement sur le marqueur et de l'afficher dans une info bulle.

Exemple de filtre avec vitesse :

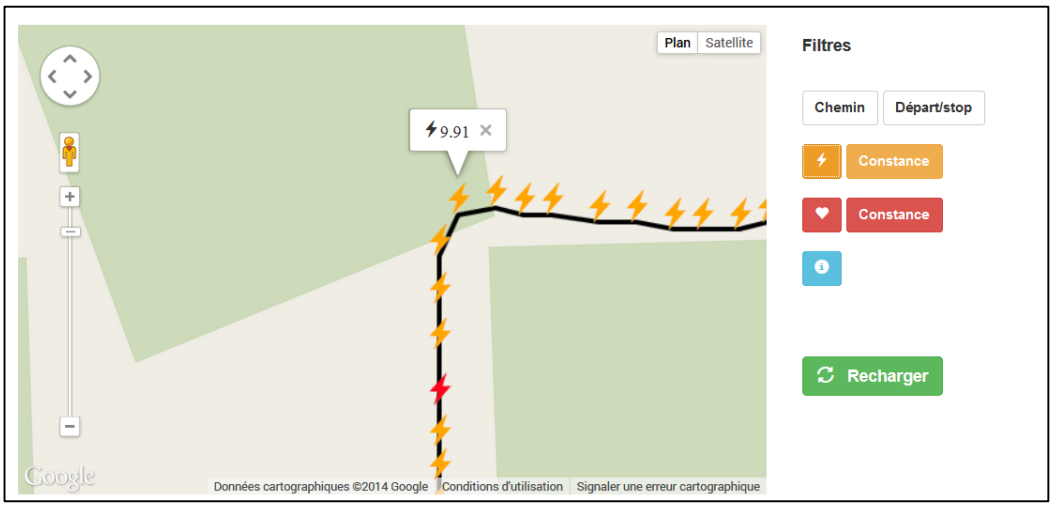


Figure 18: Page entraînement, présentation des vitesses sur trajet

En cliquant sur , un point de vitesse apparait sur toutes les zones du trajet. Le sportif a déjà un aperçu des zones du trajet où il a été le plus rapide et où il l'a été le moins. En cliquant directement sur l'indicateur de vitesse, une info-bulle présente à l'utilisateur la valeur de vitesse à ce point précis d'enregistrement. Il en va de même pour les données de pulsations.

En ce qui concerne toutes les autres données, j'ai préféré les incorporer dans une info-bulle générale qui donne toutes les valeurs de la séance au point de chaque enregistrement. J'ai favorisé la représentation des vitesses et pulsations par des couleurs différentes pour ainsi donner très vite l'information au sportif, contrairement aux données respiratoires où la couleur n'était pas nécessaire.

Pour varier la représentation et donner une autre vue pour l'utilisateur, j'ai cherché un autre moyen de représenter les écarts de vitesses et pulsations. Le moyen était d'utiliser une fonctionnalité qui s'appelle « Heat map ». Cette technique est utilisée pour afficher, par exemple, des zones où l'on trouve le plus de mouvement de population ou d'autres activités en relation avec le site.

Exemple de « Heat Maps » :

Source° : https://developers.google.com/maps/documentation/javascript/examples/layer-heatmap

Cette technique est utile pour montrer les fortes variations de vitesses ou de pulsations. Bien évidemment, le sportif peut déjà voir ces différences avec les marqueurs de couleurs variées, mais cela laisse une autre option de visualisation.

Exemple de filtre avec constance :

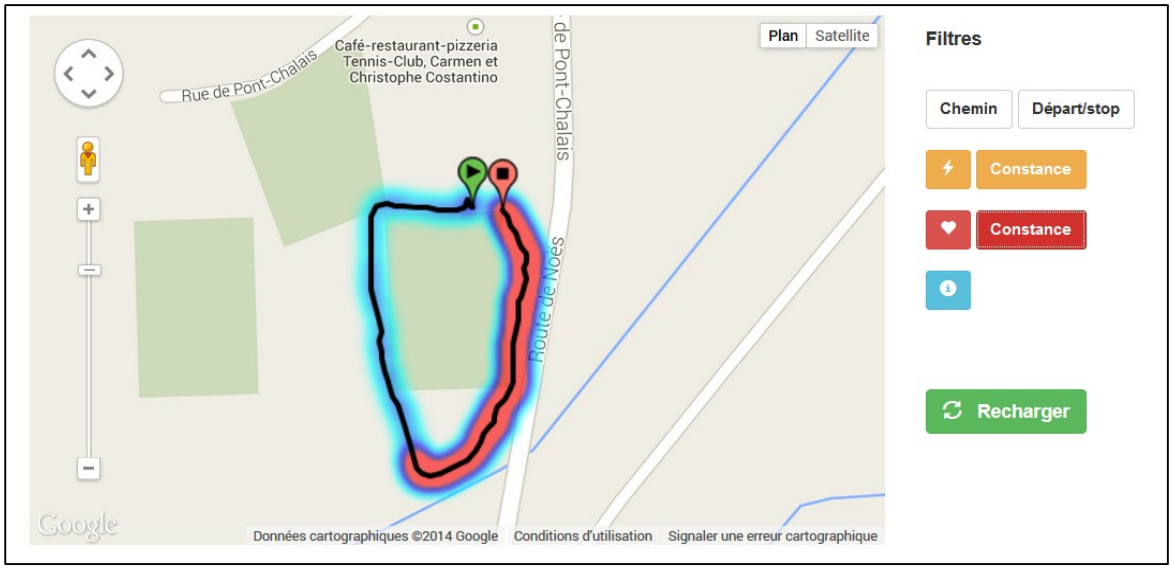


Figure 19: Page entraînement, présentation constance des pulsations

On voit ici que les pulsations vers la moitié du trajet ont fortement augmenté. Cela, s'explique par le fait que pendant mon entrainement, j'ai tenu une moyenne de 10 km/h et vers le milieu du trajet mes pulsations ont sensiblement évolué.

Exemple avec tous les détails :

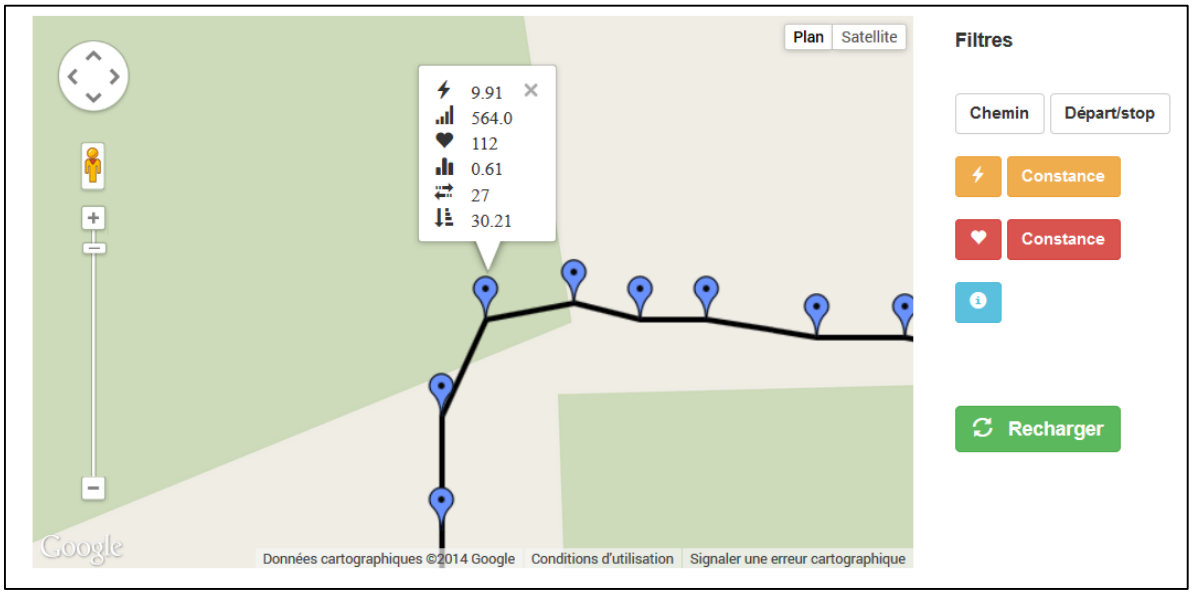


Figure 20: Page entraînement, présentation de toutes les données

L'info-bulle présente toutes les données (Android et vêtement) enregistrées à ce point précis de localisation.

Autre filtres :

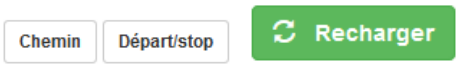


Figure 21: Filtres

L'utilisateur a la possibilité de masquer le trajet ou les points de départ et stop pour mieux visualiser l'analyse de la séance. Il peut également réinitialiser la carte et donc effacer tous les filtres rajoutés précédemment.

Légende

Pour aider le sportif à comprendre les éléments de la carte une légende a été ajoutée pour expliquer clairement la définition de chaque objet.

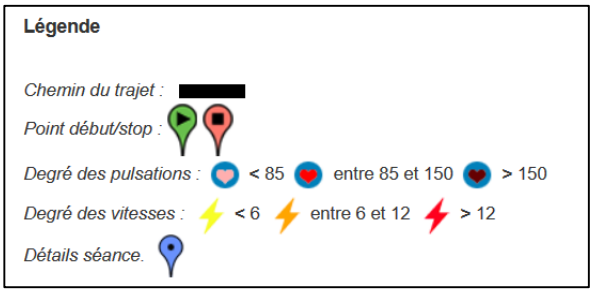


Figure 22: Page entraînement, présentation de la légende

Graphique

Pour une analyse complète un graphique comprenant toutes les données de la séance est présenté à l'utilisateur :

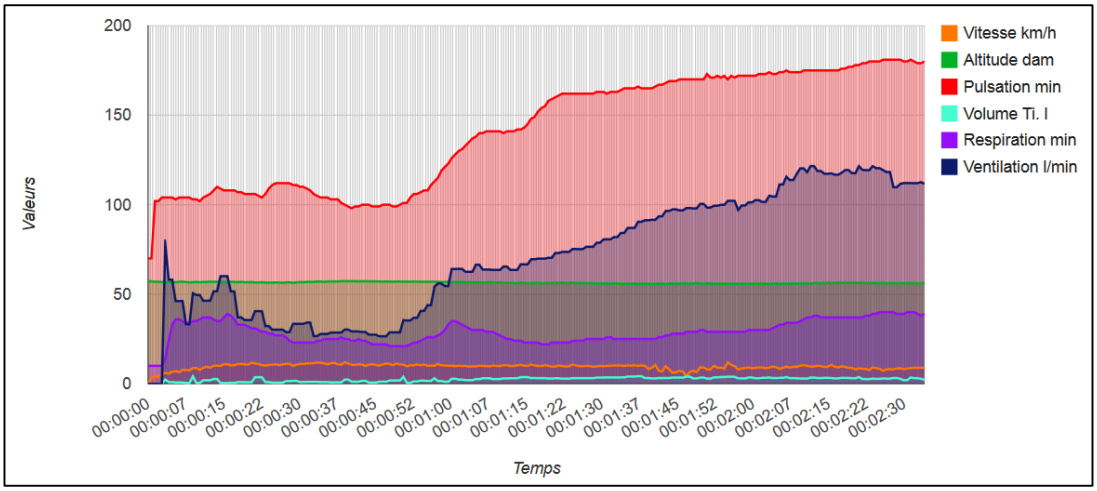


Figure 23: Page entraînement, présentation du graphique

Le sportif voit toutes les données de l'entraînement sur une échelle de temps/valeurs. L'altitude est montrée en décamètre. Cette option est préférable puisque l'altitude en mètre écrasait les autres données et qu'elles n'étaient, dès lors, plus visibles.

L'utilisateur peut cacher certaines séries du graphique pour avoir une meilleure vue sur les ou la série(s) concernée(s) :

Figure 24: Page entraînement, présentation des filtres du graphique.

4.2.3 Comparatif de séance

Sur cette page, l'utilisateur pourra comparer deux séances de sport par date de réalisation.

Premièrement, le sportif doit choisir deux dates de réalisation. Si aucune date n'est encore stipulée par celui-ci le site affiche la dernière séance.

Figure 25: Page comparer séance, présentation choix date séance

Si l'on oublie de choisir deux dates un message informera l'utilisateur de son erreur. Quand il aura choisi les deux dates, il n'aura plus qu'à cliquer sur « afficher » pour continuer.

Données

Après validation des deux dates et rafraîchissement de la page, les deux séances apparaissent. Les premières informations sont les données enregistrées d'Android en vert et du vêtement en bleu. On a donc la première séance à gauche et la deuxième à droite :

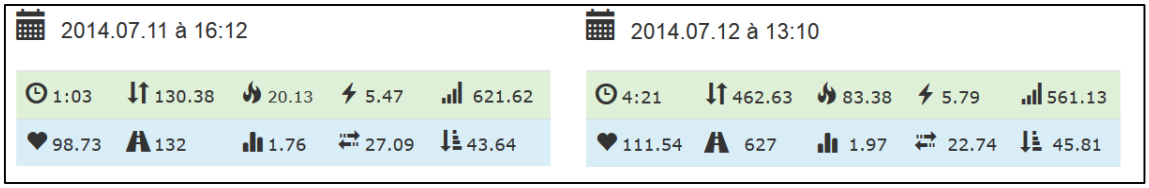


Figure 26: Page comparer séance, présentation des données de chaque séance

Pour aider le sportif à analyser les performances et différences de ces deux entraînements, une formule va calculer et montrer à l'utilisateur automatiquement là où il a été le plus performant. Il est donc préférable d'analyser deux trajets plus ou moins identiques.

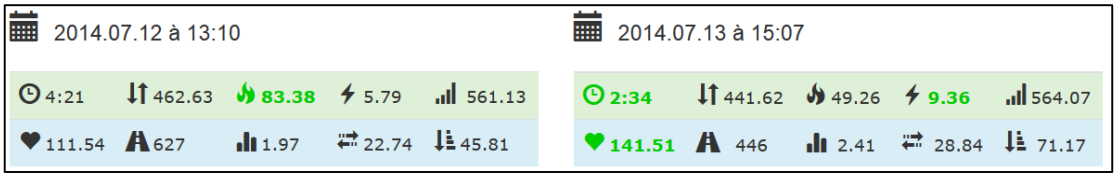


Figure 27: Page comparer séance, présentation performance

On aperçoit ici en vert les données qui sont considérées comme meilleures. Un trajet accompli avec un temps plus court est montré comme performant.

Graphiques

Comme pour la page index, des graphiques sont présentés pour analyser les différences de performance.

J'ai décidé de séparer les données en deux graphiques pour chaque séance pour une meilleure représentation. On a donc les données de vitesses/altitudes et de capacité thoracique :

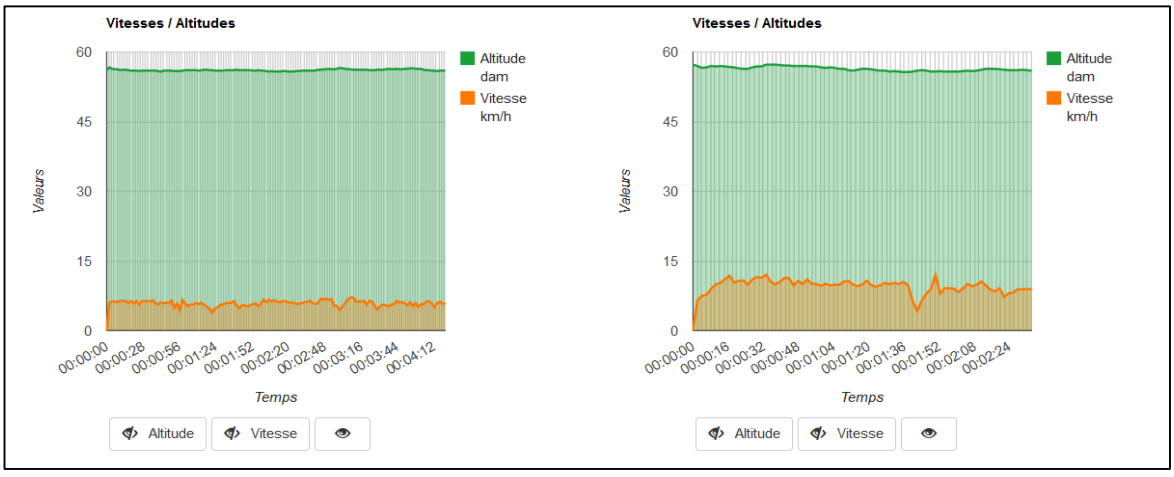


Figure 28: Page comparer séance, présentation graphique vitesse/altitude

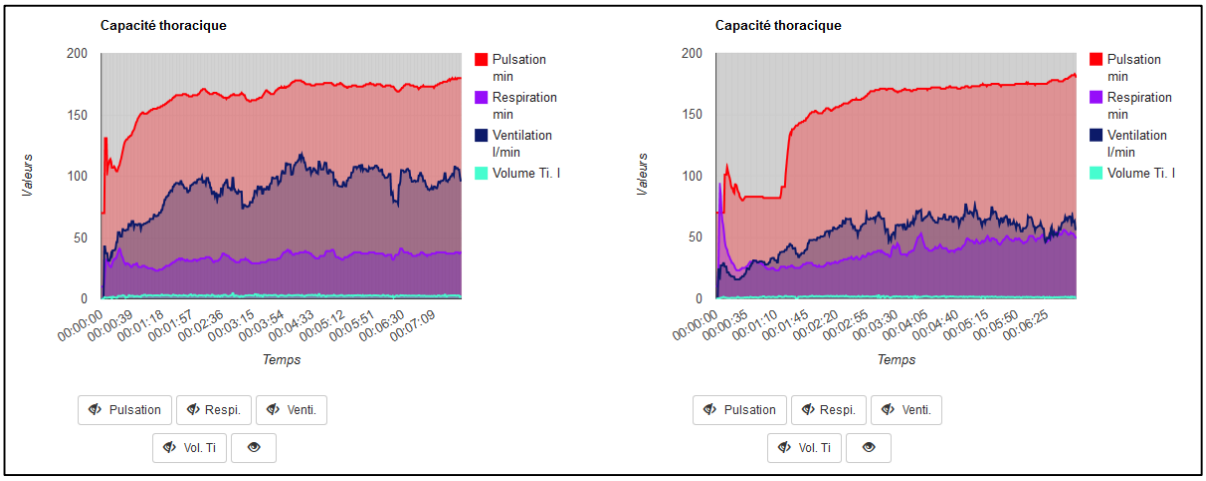


Figure 29: Page comparer séance graphique capacité thoracique

Ici les axes des graphiques sont les mêmes soit temps/valeurs. L'utilisateur peut également cacher certaines séries pour une meilleure visualisation.

Cartes

Contrairement à la page « entraînement » ici l'on va montrer les deux trajets sur la carte et identifier les différences de valeurs entre le premier et le deuxième entraînement directement sur le plan.

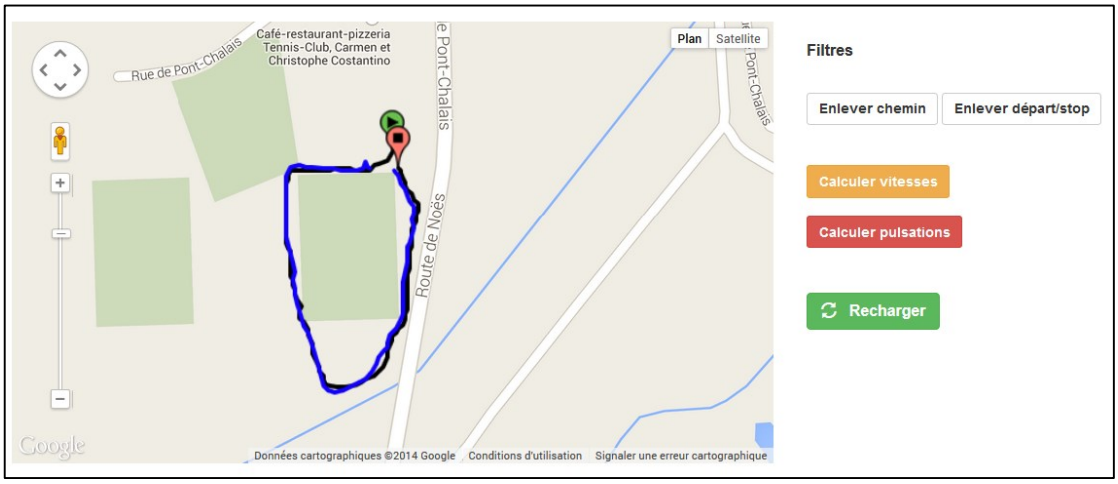


Figure 30: Page comparer séance, présentation des deux trajets sur carte

Sur la carte on voit maintenant deux trajets : un tracé noir et un bleu qui correspondent aux deux séances effectuées.

Filtres

Les filtres ici sont changés. Ils vont calculer les différences entre le premier et le second trajet de vitesses et de pulsations.

Exemple avec pulsation :

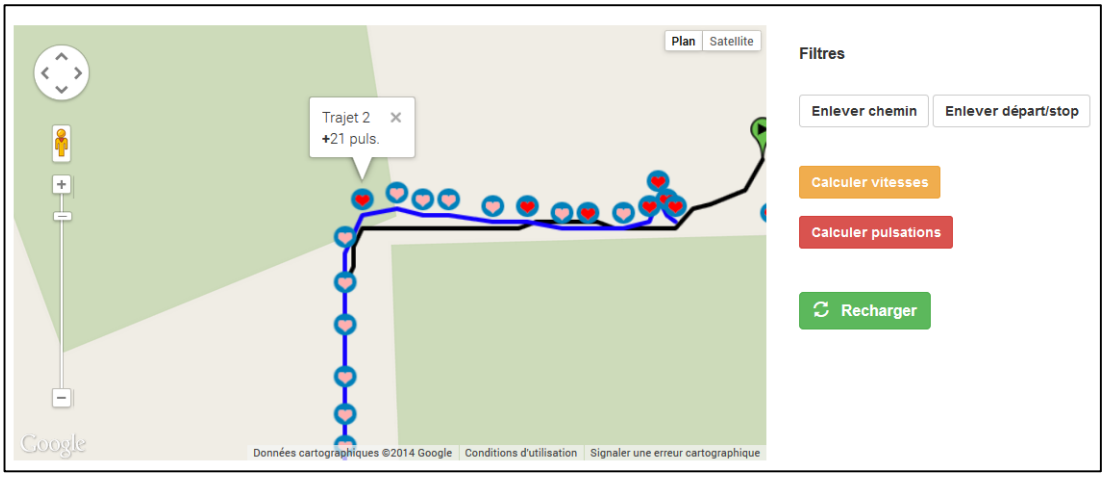


Figure 31: Page comparer séance, présentation des différences de pulsations

Ici le principe est le même sauf que maintenant la donnée de pulsation va se situer là où il y a eu augmentation. Sur l'image ci-dessus, il apparaît qu'à ce moment de la séance, le deuxième trajet avait 21 pulsations de plus que le premier. Le sportif peut donc ainsi précisément voir où il y a eu des différences.

4.2.4 Historique

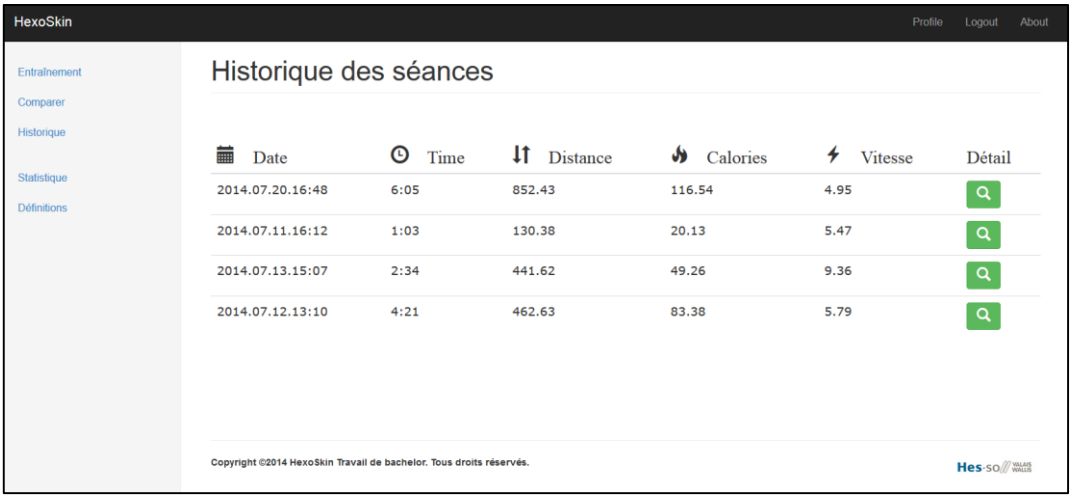


Figure 32: Page historique, présentation des toutes les séances effectuées

Cette page présente à l'utilisateur un historique de toutes les séances réalisées. L'utilisateur a un aperçu de chacune d'elles avec les diverses valeurs comme le temps, la distance et la vitesse.

Si l'utilisateur souhaite avoir plus d'informations sur la séance, comme par exemple connaître les valeurs du vêtement ou le trajet parcouru, il peut, en cliquant sur le bouton « détail », connaître ces données. Il sera redirigé sur la page « entraînement » avec la séance qu'il aura choisi de voir en détail.

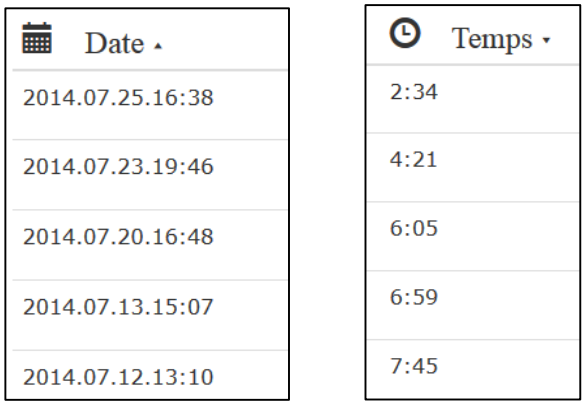


Figure 33: Page historique, présentation des filtres sur tableau

Un filtre a été ajouté pour classer les colonnes du tableau pour que l'utilisateur puisse trouver plus facilement sa séance et également voir lesquelles ont été les plus longues en terme de temps ou autre.

4.2.5 Profil

Une page de profil a été intégrée pour permettre à l'utilisateur de changer ses informations d'entraînement comme le sexe, le poids et l'âge qui ont un impact direct sur les calculs des données pendant une séance.

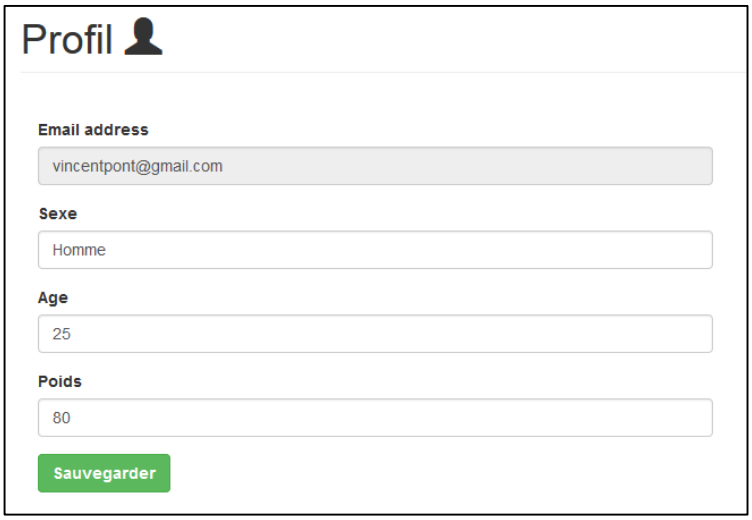


Figure 34: Page profil

Le sportif peut modifier les informations à souhait. Une fois qu'il les a modifiées il n'a plus qu'à les sauvegarder.

4.2.6 Statistiques

En guise de réflexion, j'étais à la recherche d'une page qui pourrait par exemple, nous dire rapidement le nombre d'entraînements totaux que le sportif a effectués ou le total des pas de toutes les séances réalisées. J'ai donc pensé à cette page qui analyserait tous les paramètres des séances et en ferait ressortir des statistiques générales.

4.2.7 Définitions

Certains termes scientifiques comme par exemple le volume tidal n'était pas compréhensible. C'est sur cette page que quelques définitions de terme sportif on était ajoutées. On y trouve une définition et un exemple concret des valeurs que pourrait avoir un être humain au repos et pendant l'effort.

5 Google Cloud

L'un des objectifs de ce travail de Bachelor était de sauvegarder les données de mes applications dans le Cloud. Ainsi, les données étaient disponibles partout et en tout temps.

J'ai choisi le Datastore[42] de Google car il est le plus utilisé et le plus fiable. De plus, il est gratuit et possède une large documentation et référence.

5.1 Google api explorer

C'est dans cette « console » que j'ai déployé les méthodes accessibles depuis mon application Android et web.

J'ai pu également tester mes méthodes directement sur cette plateforme pour voir si elle fonctionnait, à la suite de quoi j'ai consulté dans le Datastore si mes données avaient été sauvegardées avec succès. On y trouve également une page « logs » qui nous affiche les erreurs et les causes possibles.

6 Partie technique

Dans cette partie une présentation un peu plus technique avec du langage de programmation sera montrée et expliquée pour chacune des applications ainsi que les méthodes sur l'explorer Api de Google.

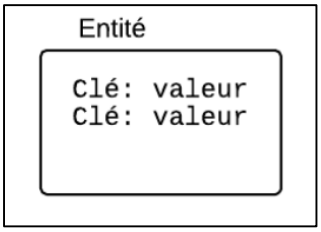
6.1 Datastore

Ce Cloud est défini en NoSQL[43]. Je l'ai choisi car il est gratuit et répondait à mes besoins.

6.1.1 Les entités

Les entités sont des regroupements d'une ou plusieurs paires clé-valeur. Elles ressemblent à ceci[44] :

[42] Glossaire :Datastore
[43] Glossaire : NoSQL
[44] Source : http://fr.openclassrooms.com/informatique/cours/montez-votre-site-dans-le-cloud-avec-google-app-engine/le-datastore



Source : http://fr.openclassrooms.com/informatique/cours/montez-votre-site-dans-le-cloud-avec-google-app-engine/le-datastore

Voici la liste des entités qui composent mon Datastore avec un exemple :

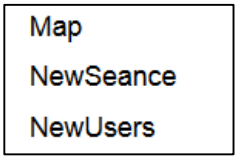


Figure 35: Liste des entités

NewUsers Entities

NAME/ID	Age	Email	Sexe	Weight
id=5688424874901504	35	meta@gmail.com	homme	120
id=5754258133614592	25	vincentpont@gmail.com	Homme	80

Figure 36: Composition de l'entité NewUsers

6.2 Appengine

Pour déployer mon application sur l'Appengine[45], j'ai utilisé Maven[46]. Maven est un logiciel libre et gratuit pour la gestion et l'automatisation de production d'application en Java.

Dès que l'on crée son code, il suffit de le valider en premier lieu avec la commande *« mvn clean install »*. Une fois ces démarches effectuées, j'ai pu déployer sur l'Appengine avec *« mvn appengine :update »*.

[45] Glossaire : Appengine
[46] Glossaire : Maven

La dernière opération était de générer la librairie qui contenait mes méthodes avec *« mvn appengine:endpoints_get_client_lib ».* Cette librairie et indispensable pour l'appel des méthodes du côté de l'application Android.

Après intégration de celle-ci dans mon application Android, je pouvais appeler les méthodes créées précédemment.

6.3 Les méthodes

Les méthodes présentées ici sont les méthodes déployées directement sur l'Appengine de Google. Par la suite, je fais appel à ces méthodes soit du côté Android soit du côté web.

6.3.1 Méthodes côté Android

putDataSeance

Cette méthode permettait de sauvegarder toutes les données enregistrées pendant la séance de sport comme les calories brûlées, la distance, la vitesse moyenne. J'ai également rajouté certaines données qui m'ont semblé pertinentes et indispensables comme l'email et la date de l'utilisateur.

putDataMap

Cette méthode enregistre toutes les données de localisation comme la latitude et longitude. Elle enregistre également la vitesse et l'altitude à chaque point de localisation.

putUsers

Celle-ci sauvegarde l'utilisateur et ses données comme le sexe, l'âge et le poids.

UserExist

Cette méthode me renvoie une valeur vraie ou fausse si le sportif est déjà enregistré, cela permet ainsi de ne pas sauvegarder les données encore une fois.

6.3.2 Méthodes côté web

getDataMap

Renvoie toutes les donnée de localisation pour pouvoir ainsi recréer la carte telle quelle était présentée dans Android. Les paramètres sont l'email et la date.

getAllWorkoutDates

Renvoie toutes les dates des entraînements comprenant l'email passé en paramètre.

getDataWorkoutByEmail

Renvoie tous les entraînements comprenant l'email passé en paramètre.

getDataWorkoutByEmailAndDate

Renvoie un seul entraînement. Les paramètres d'entrée sont l'email et la date.

getUsers

Renvoie toutes les informations de l'utilisateur. Le paramètre d'entrée est l'email.

updateUser

Met à jour les informations de l'utilisateur.

6.4 Partie application Android

Ici sera expliqué quelques parties de code important comme l'activité « carte » et l'activité « résumé séance » qui présentent et sauvegardent les données dans le Cloud.

6.4.1 Login

La partie login/logout est entièrement géré par Google. Il a suffi de générer une class login et par la suite d'intégré à mon application et de faire le lien entre les différentes activités. C'est pourquoi je ne vais pas rentrer dans les détails pour cette partie.

6.4.2 Carte

La partie qui nous intéresse ici est de savoir comment les données de localisation sont reçues et sauvegardées.

Les données du GPS sont reçues par l'intermédiaire d'un *Listener*[47]. C'est-à-dire que chaque fois que la position changeait, je le savais. Je pouvais ainsi par exemple sauvegarder les données comme la latitude ou la vitesse à ce moment.

Ci-dessous la méthode *onLocationChanged* qui passe l'objet location en paramètre et qui nous permet par la suite de sauvegarder ce nouveau point de localisation à chaque changement de position est relevée :

```
public LocationListener locationListener = new LocationListener() {

    public void onLocationChanged(Location location) {

        locations = location;
        longitude = location.getLongitude();
        latitude = location.getLatitude();
```

Figure 37: Code, Listener sur GPS

Par la suite, en se positionnant toujours dans cette méthode, je sauvegardais ma latitude et longitude dans des listes. J'ai fait de même avec toutes les autres données de ma séance :

```
// Add to the list the new Lat & Long
listLat.add(latitude);
listLong.add(longitude);
```

Figure 38: Code, ajout position dans list

Après cela, je devais dessiner le trajet ainsi que le marqueur de départ et de fin pour montrer le trajet au sportif. J'ai utilisé l'objet « Polyline » qui par le biais de deux points de localisation va dessiner un trait qui part du point A au point B :

[47] Glossaire : Listener

```
// Draw path
// Add a point(location) and draw the polyline when 2 points are inserted
rectOptions.add(new LatLng(latitude, longitude));
mMap.addPolyline(rectOptions);
```

Figure 39: Code, création chemin sur carte

```
mMap.addMarker(new MarkerOptions().position
        (new LatLng(listLat.get(0), listLong.get(0))).title("Start"));
```

Figure 40: Code, ajout marqueur départ/fin pour trajet

Pour la vitesse et l'altitude, c'est directement l'objet *location* qui me donnait ces informations :

```
// Get altitude and add to list
altitude = locations.getAltitude();
listAltitude.add(altitude);
```

Figure 41: Code, ajout altitude dans une list

```
locations.getSpeed()*3.6)
```

Figure 42: Code, ajout vitesse en km/h

Malheureusement pour l'altitude la précision laisse à désirer, ce problème étant lié directement au GPS et non à mon code. La vitesse est multipliée par 3.6 pour avoir des km/h.

La précision du GPS pouvait être réglée par le biais de l'objet *LocationManager :*

```
// Launch listener GPS, 3000 = time until update in second, 3 = meters until update
locationManager.requestLocationUpdates(bestProvider, 3000, 3, locationListener);
```

Figure 43: Code, précision GPS

Pour le calcul des calories brûlées, j'ai dû faire des recherches pour connaître la formule.

Voici les formules que j'ai utilisées :

Calcul homme

Calories brûlées = [(Age x 0.2017) + (Poids x 0.09036) + (Pulsation x 0.6309) -- 55.0969] x Temps / 4.184.

Calcul femme

Calories brûlées = [(Age x 0.074) -- (Poids x 0.05741) + (Pulsation x 0.4472) -- 20.4022] x Temps / 4.184.

N'ayant pas directement les pulsations, j'ai dû faire une moyenne des battements du cœur par minute que pourrait faire un adulte.

220-age

Pour la distance parcourue, l'utilisation d'une méthode de l'objet *location* me permettait, en passant en paramètre deux points de localisation, de connaître le nombre de mètres qui les séparaient :

```
distance += locationA.distanceTo(locationB);
```

Figure 44: Code, calcul de la distance

6.4.3 Résumé séance et sauvegarde

Ici, je vais présenter les méthodes de la librairie que j'ai générée depuis Maven :

Figure 45: Librairie des méthodes générées depuis Appengine

C'est dans cette libraire que toutes mes méthodes sont déclarées. Voici un exemple de la façon dont j'ai sauvegardé mes données de localisation en appelant la méthode *putDataMap :*

```
// Put the data of the map in Datastore
putDataMap = new AsyncTask<Void, Void, PutDataMap> () {

    @Override
    protected PutDataMap doInBackground(Void... voids) {

        // Retrieve service handle.
        Helloworld apiServiceHandle = AppConstants.getApiServiceHandle();

        try {
            // Call the api method and pass the values to save the data
            PutDataMap putlistLat = apiServiceHandle.greetings()
                    .putDataMap(sdfDataStore.format(date),
                            emailUser, listStringLat, listStringLong,
                            listStringSpeed, listStringAlti);

            putlistLat.execute();

        } catch (IOException e) {
            Log.e("Error", e.toString());
        }
        return null;
    }
};
```

Figure 46: Code, exemple appel méthode puDataMap

Ci-dessus on voit que je fais appel à une méthode *AsyncTask*[48] qui est indispensable pour faire nos appels aux méthodes. J'appelle un peu plus bas la méthode *putDatMap* et je lui passe en paramètre les listes de latitudes et longitudes ainsi que les listes de vitesses et d'altitudes en rajoutant la date et l'email du sportif.

Il ne me reste plus qu'à exécuter cette méthode par exemple dans un bouton *listener* :

```
// Listener save data in DataStore
imageButtonSave.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {

        // Put data in datastore
        putDataSeance.execute();
        putDataMap.execute();
        putUsers.execute();
        Toast.makeText(getApplicationContext(), "Seance saved.",
                Toast.LENGTH_SHORT).show();
        startActivity(intentInfos);
    }
});
```

Figure 47: Code exemple d'exécution méthode AsynchTask

Voilà les données sont envoyées et traitées par le Datastore.

[48] Définition de la classe : http://developer.android.com/reference/android/os/AsyncTask.html

6.5 Partie application web

La majorité du code est composé de JavaScript. En effet, l'utilisation de la carte et des graphiques de Google est entièrement fait en JavaScript. Toutes les données qui vont servir à construire ceux-ci sont récupérées depuis Java et après quelques modifications sont retransmises du côté client pour affichage et présentation.

6.5.1 Hexoskin api

L'api d'Hexoskin est en Rest et les données récupérées sont sous format Json. Il m'a fallu donc depuis des classes Java, faire des requêtes Rest pour interroger cette api et récupérer les informations dont j'avais besoin.

Récupération

Premièrement pour récupérer les données du vêtement, je devais connaître au préalable la séance à interroger par sa date de réalisation. Par la suite, j'obtenais un identifiant de cette séance.

Après récupération de cet identifiant, je pouvais faire des requêtes pour savoir par exemple toutes les données de pulsations qui ont été enregistrées pendant cet entraînement. Voici un exemple de données de pulsations :


```
[{"data": {"19": [[359902025470, 70], [359902025726, 70], [359902025982, 70], [359902026238, 64],
[359902027262, 77], [359902027518, 77], [359902027774, 77], [359902028030, 77], [359902028286, 73]
```


Figure 48: Exemple données retournées depuis l'api Hexoskin (Pulsations)

Ci-dessus les deuxièmes valeurs représentent les pulsations enregistrées. On voit que la première valeur était donc septante et la dernière septante-trois. La première valeur est le « epoch timestamp » qui me donnait le total des secondes depuis le premier janvier 1970.

Après avoir récupéré ces informations, je les passais dans des listes pour qu'elles soient plus faciles de manipulation afin de les ajouter dans mes graphiques.

Je ne vais pas montrer ici d'exemples car le code est un peu fastidieux pour arriver à ces résultats. Cependant vous pourrez bien évidemment prendre connaissance de tout cela directement dans mes classes Java où toutes les méthodes sont commentées.

Affichage

Après récupération des informations, je devais construire mes graphiques ou faire des moyennes pour présenter par exemple au sportif la moyenne des pulsations de l'entraînement.

La première opération consistait à récupérer ma liste qui contenait toutes les données de pulsations. Ensuite je faisais une moyenne de toutes ces valeurs à l'aide d'une méthode :

```
List<String> listPulsation = restHEXO.returnAllValueFromJson(dateHEXO, "19");
```

```
<TD title="Pulsation min moyenne." class="info">
    <span  style="font-size:21pt;" class="glyphicon glyphicon-heart"></span>
    <span style="font-size:14pt; font-family:Verdana;">
    <% out.print(restHEXO.getAverageFromList(listPulsation));  %> </span>
</TD>
```

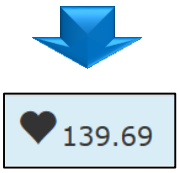


Figure 49: Code, affichage de la moyenne de pulsation

Pour ce qui était des données dans les graphiques ou la carte, le principe était plus ou moins le même. Cependant, je devais passer ces valeurs directement dans des variables JavaScript. J'ai été confronté à quelques problèmes lors de la création des graphiques et de la carte qui vous seront décrits plus bas dans la section « problèmes rencontrés ».

6.5.2 Datastore et donnée Android

Les données Android étant sauvegardées dans le Datastore de Google, je devais les prendre pour les afficher sur la plateforme web. Les appels pour interroger le Datastore ont été également faits par des requêtes Rest.

Récupération

Voilà les données telles qu'elles s'affichent dans le Datastore :

NewSeance Entities

NAME/ID	Calories	Date	Distance	Email	Speed	Time
id=5099593180119040	116.54	2014.07.20.16:48	852.43	vincentpont@gmail.com	4.95	6:05

Figure 50: Entité NewSeance

Affichage

Et voici un exemple pour l'affichage des calories totales brûlées durant un entraînement :

```
listWorkout = rest.getDataWorkoutByEmailAndDate(dateToShow,
        "vincentpont@gmail.com");
```

```
<TD title="Calories brûlées." class="success">
    <span style="font-size:21pt;" class="glyphicon glyphicon-fire"></span>
     <span style="font-size:14pt; font-family:Verdana;"> 
      <% out.print(listWorkout.get(3)); %></span>
</TD>
```

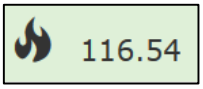


Figure 51: Code, exemple affichage calories brûlées total

6.6 Twitter Bootstrap

La partie design a été réalisée avec l'aide de Bootstrap. Il s'agit d'un outil d'aide à la conception de site web et application web. Cet ensemble contient notamment du code Html, Css et JavaScript.

Étant donné le délai imparti pour ce travail de Bachelor, je n'avais que peu de temps à consacrer au design de mon site. Cet outil m'a permis de gagner en efficacité.

Il possède une large gamme de classes, d'icônes et de style différents qui m'ont donné l'occasion d'approcher un visuel plus professionnel à ma plateforme et de lui offrir une bonne ergonomie.

6.7 Tests unitaires

Lors de mon développement, j'ai consacré une partie à la réalisation de tests unitaires. Ces tests m'ont aidé à debugger plus facilement et de plus, ils m'ont fait réaliser que certaines de mes méthodes comprenaient des erreurs que je n'avais pas remarquées avant de les tester.

J'avais une méthode qui faisait une moyenne et j'avais une variable « count » qui comptait le nombre d'apparitions des valeurs. Par la suite, je pus, les diviser avec cette même variable. J'avais initialisé cette variable à un pour éviter une division par zéro s'il n'y avait aucune données. Le problème, c'est que mon count commençait à s'incrémenter à un et donc le résultat était faussé. Je ne m'en suis aperçu qu'en faisant ces tests unitaires.

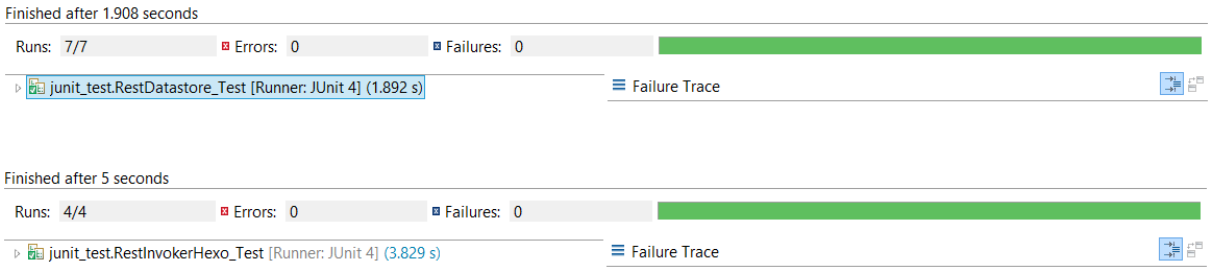


Figure 52: Affichage de la réussite des 11 tests unitaires

Mes tests unitaires ont été implémentés sur les deux classes qui invoquent les données soit le Datastore et l'Api Hexoskin qui font des appels en Rest. J'ai donc testé les méthodes les plus importantes pour voir si elles fonctionnaient et lors de bug[49], le ciblage de problèmes fut plus rapide.

6.8 Logiciel de gestion de version

Pour mon travail de Bachelor, j'ai utilisé un gestionnaire de code source pour pouvoir sauvegarder mon code et ainsi ne rien perdre.

[49] Glossaire : Bug

J'ai choisis le logiciel GitHub car je l'avais déjà utilisé durant les projets du 5ème et 6ème semestre et j'étais plus ou moins satisfait de celui-ci. Il est très facile d'utilisation et dispose d'une interface web pour consulter son repository[50].

Notre code source est public et peut être vu par n'importe qui. Si l'on ne veut pas autoriser cela, il faut malheureusement passer à une version payante mensuelle qui varie selon le nombre de repository souhaités.

7 Méthodologie agile

Il a été décidé, au début du projet, d'adopter une méthodologie agile. C'est-à-dire de tenir un Product Backlog[51] à jour et de simuler par le biais de Monsieur Schumacher et Monsieur Alves les rôles de Product Owner[52] et de client.

J'ai tenu ainsi un développement itératif composé de sprints. Un sprint durait environ 2 semaines. À chaque rendez-vous le Product Owner décidait des user stories à implémenter pour la fin du sprint en tenant compte, bien sûr, des exigences et besoins du client et acceptait ces dernières.

8 Difficultés rencontrées lors du développement

8.1 Application Android

Lors de mon développement sur Android, je n'ai pour ainsi dire pas rencontré de problèmes majeurs qui m'ont stoppé dans mon développement. Cependant, voici quelques confrontations que j'ai eues :

[50] Glossaire :Repository
[51] Glossaire : Product backlog
[52] Glossaire : Product owner

8.1.1 Carte et affichage

L'un des challenges était surtout d'intégrer la carte de Google et de faire en sorte que l'affichage de celle-ci soit simple avec une bonne ergonomie. Je dus réfléchir comment présenté les valeurs de la séance sans pour autant masquer la carte en arrière-plan. Avec des techniques de style comme l'opacité, je pus présenter tous les éléments correctement à l'utilisateur.

Le second objectif était de pouvoir sauvegarder les données de la séance sur Google Cloud et particulièrement le Datastore. Je dus comprendre comment on y avait accès depuis Android et comment les données étaient sauvegardées.

8.1.2 Restriction du Datastore

Un jour, pendant que je testais mon application, je m'aperçus que les données ne se sauvegardaient pas quand je commençais à avoir beaucoup d'informations. Les informations auxquelles je fais allusion sont les données de localisation (latitudes et longitudes) et de séance (altitudes et vitesses). Ces informations sont sauvegardées dans des listes en Java à chaque changement de direction de la part du sportif. Dès que ces données dépassaient 500 caractères, elles n'étaient tout simplement plus sauvegardées dans le Cloud.

Après plusieurs recherches et une erreur m'indiquant qu'il y avait une restriction de 500 caractères, je compris que j'étais en face d'un problème. Je devais trouver un autre moyen pour sauvegarder toutes mes listes. Je réglais plus tard ce problème en transformant mes listes de « String » en liste de « Double » lorsque vint le moment de les sauvegarder en tant que propriété dans le Datastore.

Malheureusement, le problème était en partie résolu. Il devait également avoir une limitation pour mes listes de « Double ». En effet, cela généré une erreur « Bad request » lorsque ma séance comprenait trop de donnée.

J'ai donc trouvé deux moyens pour contrer voir éradiquer ce problème. Le premier étant de stipuler un temps ou mètre plus important d'intervalle de mise à jour pour le GPS. De ce fait, moins de données étaient sauvegardées et je ne rencontrais plus ce problème.

Le deuxième, est qu'il est peut être préférable de changer de Cloud et de prendre le Cloud SQL qui est moins restrictif mais payant.

8.2 Application web

L'un des problèmes majeurs que j'ai rencontré pendant mon développement sur l'application web était de coupler les données des capteurs avec les données de mon application Android dans des graphiques ou sur une carte.

Les données du vêtement sont stockées sur des serveurs Hexoskin. Pour appeler ces informations, je devais passer par leur Api qui était en Rest et retourner les données sous format Json.

Pendant la création des graphiques sur la page « entraînement », je remarquai que les données et donc la ligne directrice du graphique était faussée. Ce problème venait du fait que quand mon application Android enregistrait une donnée, le vêtement, lui, en enregistrait entre deux et trois données de plus. Et ce chiffre variait, puisqu'il pouvait y avoir par exemple deux fois et demi de données en plus dans une liste de pulsation du côté du vêtement que dans une liste de vitesse du côté d'Android.

Je dus donc réfléchir à la résolution de ce problème et surtout comment présenter au mieux les données dans les graphiques et sur la carte sans fausser les résultats de la séance.

Une solution qui m'a été proposée par Monsieur Schumacher lors d'une séance était de rajouter les données manquantes dans ma liste, par exemple de vitesse, en faisant une moyenne du premier et troisième chiffre, ce qui me donnait une estimation de ce que pouvait être le deuxième chiffre.

Cela marchait bien pour autant que je me rapprochais d'un ratio entier. En effet, lorsque le ratio du nombre de données était un chiffre à virgule comme deux et demi cela me posait un problème car dans un tableau en Java je ne peux que rajouter deux données mais pas deux données et demie.

J'ai donc fait au mieux pour représenter les données dans les graphiques à même échelles temps/valeurs et de même sur la carte.

9 Améliorations

9.1 Synchronisation des données

Le problème majeur de ce travail Bachelor était de lier mes données avec celle du vêtement qui n'était pas enregistrées au même moment ou tout simplement enregistrées différemment (format de date). Cela a eu pour conséquence une perte de précision dans mes graphiques et également quand je mappais l'effort sur la carte.

C'est pourquoi établir un lien (Bluetooth ou autre) lors de l'enregistrement des données du vêtement et de mon application et pouvoir enregistrer les données au même moment des deux côtés améliorerait grandement la simplicité du code et surtout la précision des données.

9.2 Datastore

En ce qui concerne la partie des requêtes Rest pour interroger les données sur le Datastore, il est possible de faire plus simple et différemment. En effet, l'on peut directement faire appel au Datastore avec un objet, muni de libraires comme « Objectify[53] » pour nous aider à construire nos requêtes. De plus, peut être éventuellement changer de Cloud et de prendre le Cloud SQL qui est peut-être plus adéquat pour cette application qui enregistre vraiment beaucoup de donnée simultanément.

Malheureusement, je n'ai pas eu le temps d'analyser ce changement en terme de performance ou les conséquences bénéfiques ou non qu'il pourrait y avoir. Mais je suis sûr qu'au final le code serait bien plus simplifié et mieux adapté.

[53] Glossaire : Objectify

10 Nouvelles idées

J'ai eu quelques idées qui pourraient être implémentées dans un développement futur.

10.1 Images

Ceci est peut-être plus une idée pour le vêtement mais qui dépendrait également du site : le fait de pouvoir à un moment donné prendre une photo avec le vêtement, donc que le vêtement soit muni d'une petite caméra qui, quand le sportif le souhaite, peut prendre une photo. Ainsi sur la plateforme web, on visualiserait des points sur le trajet où l'on pourrait cliquer pour voir le cliché que le vêtement a pris. Alors l'on se rendrait plus facilement compte par exemple qu'à ce moment du trajet on a une montée assez pentue.

Cette photo permettrait une meilleure visualisation du trajet pour des autres sportifs qui ne connaissent pas cet itinéraire pour en connaître la difficulté directement en images. L'on pourrait aussi peut être intégrer des petites portions de vidéos à la place des images.

10.2 Système de hauts faits

J'entends par hauts faits une sorte de système de récompense qui donne des médailles ou badges à un sportif de manière virtuelle et qui seraient affichés sur son profil. Par exemple, lorsque le sportif atteint 10 entraînements, il reçoit une médaille correspondant à ce haut fait. Cela pourrait motiver davantage le sportif à pratiquer du sport et à utiliser la solution web. On pourrait après cette implémentation avoir un top 100 des meilleurs sportifs sur le site, classés par différents critères. Au final, chaque utilisateur pourrait consulter le profil de n'importe quel utilisateur comme un réseau social.

11 Conclusion

Premièrement, ce projet très enrichissant m'a apporté de multiples satisfactions et expériences dans le domaine de l'informatique et notamment en Java EE, Android et sur les nombreuses fonctionnalités offertes par Google comme son Cloud.

Cela m'a apporté également aussi des connaissances dans le domaine du sport, notamment sur le calcul des calories brûlées lors d'une séance ou simplement sur des termes scientifiques par exemple comme le volume tidal.

Ce vêtement peut vraiment être utile pour un sportif et apporte une grande plus-value surtout si celui-ci pratique du sport à haut niveau. J'ai eu l'occasion d'analyser ma séance sur toutes ses possibilités. Le fait d'avoir apporté ma solution en terme de géolocalisation et de montrer un plan et des graphiques a révélé une meilleure visualisation de l'entraînement.

J'aurai souhaité, pouvoir présenter les données de manière plus précise. Compte tenu du manque de synchronisation du vêtement et de mon application Android cela ne m'a pas permis de parvenir totalement à une précision optimale souhaitée.

J'espère avoir atteint au mieux les objectifs de ce travail de Bachelor et avoir su mettre en pratique les compétences acquises pendant ces 3 années à l'HES-SO de Sierre.

Sources

Différences entre application natives et web app, forces et faiblesses (par **Rédacteur Invité** le 17 janvier 2013). Consulté le 02.05.2014, lien :

http://www.journaldugeek.com/2013/01/17/applications-natives-contre-les-webapps-forces-et-faiblesses/

Natif vs web app, quel est le bon choix (par **Priya Viswanathan**). Consulté le 02.05.2014, lien : http://mobiledevices.about.com/od/additionalresources/a/Native-Apps-Vs-Web-Apps-Which-Is-The-Better-Choice.htm

Différence entre PhoneGap et autres framework (par **Florent Lamoureux** le 01.02.2012). Consulté le 03.05.2014, lien :

http://www.florentlamoureux.fr/blog/differences-entre-phonegap-appcelerator-titanium-sencha-touch-et-jquery-mobile/

PhoneGap: The Good, The Bad and The Weird (par **Jairo Pineda** le 29.07.2014). Consulté le 03.05.2014, lien :

http://www.rebusagency.com/phonegap-the-good-the-bad-and-the-weird.html

Api Phonegap. Consulté le 03.05.2014, lien :

http://docs.phonegap.com/fr/1.3.0/phonegap_geolocation_geolocation.md.html

Mobile: Native Apps, Web Apps, and Hybrid Apps (par **Raluca Budiu**, le 14.09.2013). Consulté le 03.05.2014, lien :

http://www.nngroup.com/articles/mobile-native-apps/

jQuery: the good, the bad & the ugly (par **Richard Larson**, le 03.09.2012). Consulté le 05.05.2014, lien :

http://www.webdesignerdepot.com/2012/09/jquery-the-good-the-bad-and-the-ugly/

Google Cloud Datastore documentation. Consulté le 06.05.2014, lien :

https://developers.google.com/datastore/

Google maps documentation by Google. Consulté le 15.05.2014, lien :

https://developers.google.com/maps/documentation/android/?hl=fr-FR

Géolocalisation sous Android (par **Benbourahla Nazim** 26.05.2012). Consulté le 16.05.2014, lien :

http://www.tutos-android.com/geolocalisation-android

How to Create a Google Map Application using Android Studio (2013-12-31). Consulté le 26.05.2014, lien :

http://www.tuicool.com/articles/7Jfaem

Api Hexoskin documentation. Consulté le 26.05.2014, lien :

https://api.hexoskin.com/docs/page/overview/

Google+ Sign-In pour Android by **Google.** Consulté le 10.06.2014, lien :

https://developers.google.com/+/mobile/android/sign-in

Outil pour personnalisation des boutons sur Android. Consulté le 18.06.2014, lien :

http://angrytools.com/android/button/

List icones sur Android. Consulté le 18.06.2014, lien°:

http://androiddrawableexplorer.appspot.com/

Google Appengine by **Google**. Consulté le 20.06.2014, lien :

https://developers.google.com/appengine/docs/java/datastore/entities

Backend API Tutorial by **Google**. Consulté le 25.06.2014, lien :

https://developers.google.com/appengine/docs/java/endpoints/getstarted/backend/

Google Endpoint by **Google**. Consulté le 25.06.2014, lien :

https://developers.google.com/appengine/docs/java/endpoints/getstarted/clients/android/

Stocker des fichiers sur Google Cloud Storage (par **Mathieu Nebra**). Consulté le 25.06.2014, lien :

http://fr.openclassrooms.com/informatique/cours/montez-votre-site-dans-le-cloud-avec-google-app-engine/stocker-des-fichiers-sur-google-cloud-storage

PHP Runtime Environment by **Google**. Consulté le 28.06.2014, lien :

https://developers.google.com/appengine/docs/php/

Using the Google Plugin for Eclipse by **Google**. Consulté le 28.06.2014, lien :

https://developers.google.com/appengine/docs/java/tools/eclipse?hl=fr

Google login sur site en JavaScript by **Google**, Consulté le 28.06.2014, lien°:

https://developers.google.com/+/web/signin/#choosing_a_sign-in_flow

Montez votre site dans le cloud avec Google Appengine (par **Mathieu Nebra**). Consulté le 01.07.2014, lien :

http://fr.openclassrooms.com/informatique/cours/montez-votre-site-dans-le-cloud-avec-google-app-engine/stocker-des-fichiers-sur-google-cloud-storage

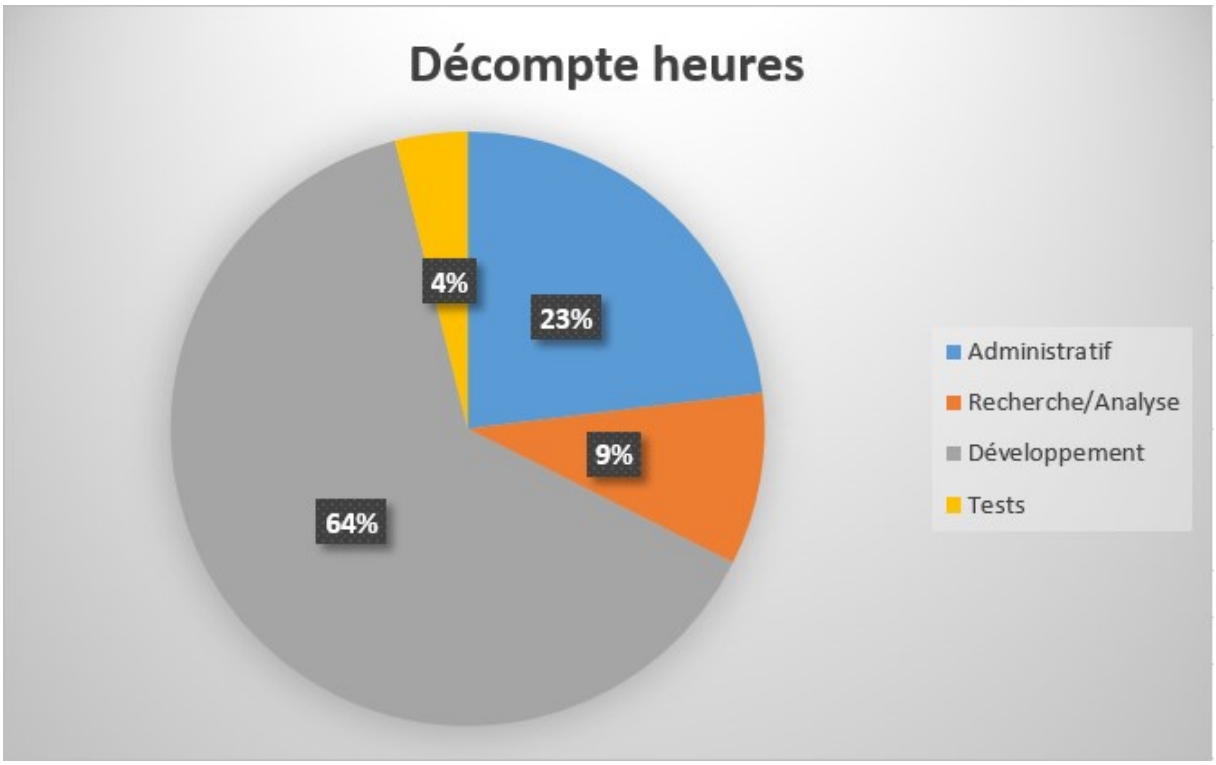
Technologie JSP (par **Médéric Munier**) Consulté le 02.07.2014, lien :

http://fr.openclassrooms.com/informatique/cours/creez-votre-application-web-avec-java-ee/la-technologie-jsp-1-2

Technologie Java EE (par **Médéric Munier**) Consulté le 02.07.2014, lien :

http://fr.openclassrooms.com/informatique/cours/creez-votre-application-web-avec-java-ee/la-technologie-jsp-1-2

Annexe I : Décompte des heures par catégories



Catégories	Heures
Administratif	85
Recherche/Analyse	34,5
Développement	234,5
Tests	14,5
Total	***368,5***

54

[54] Pour tous les détails, voir fichier *Heures.xlsx*

Annexe II : Product Backlog / User stories Android et web app

US Nr.	Theme	User Stories			,and will be validated so	Priority	Status	Story Points	Sprint
		As an/a _	I want to _	so that _					
1	Android/Connexion	Utilisateur	login sur mon application avec mon compte Google	Je possède un compte et je peux utiliser l'application	-tester que les champs de connexion apparaissent ainsi que le bouton se	1		3	1
2	Android/Données de base	Utilisateur	donner mon poids et mon age et mon sexe	Je pourrai consulter les calories total que j'ai brûlé durant la séance	-tester que les champs apparaissent, tester que les données sont sauvegardées	2		1	1
3	Android/Séance	Utilisateur	démarrer une séance de sport ou l'arrêter	Je pourrai consulter la duration total de ma séance de sport ou l'arrêter à tout moment	-tester que le chronomètre apparait et fonctionne parfaiteement et que les données sont sauvgardées	3		5	1
4	Adnroid/GPS	Utilisateur	mettre en fonction le gps	la géolocalisation est activée et je pourrais consulter la carte et le chemin parcouru par la suite	-tester que le gps se mets bien en route, tester qu'on ne peut pas démarrer une séance sans avoir activer préalablement le GPS	4		1	1
5	Android/Analyse séance	Utilisateur	voir une carte cartographié ou plan qui me montre le trajet effectué durant ma séance de sport	Je pourrai voir le trajet que j'ai effectué sur une carte (distance totale, moy km/h)	-tester que la carte apparaisse correctement, que les données GPS sont correctement insérées pour créer le chemin sur la carte	6		5	1
6	Android/Analyse séance	Utilisateur	avoir un petit résumé de ma séance de sport (Duration, calories)	Je pourrai savoir si ma séance a été performante	-tester que les données apparaissent parfaitement et que les formules soient	5		3	1
7	Adnroid/Analyse	Utilisateur	avoir des informations complémentaire comme la distance effectué ou mètre/min	Je pourrai mieux analyser ma séance de sport	-test que les valeures et calcules soient justes et s'affichent	1		3	2
8	Android/Cloud	Utilisateur	sauvegarder ma séance dans le cloud (données séances)	Je pourrai les consulter à l'avenir et elle ne seront pas perdues	-tester que les données sont bien sauvegardées dans le cloud	2		13	2
9	Android/Analyse séance	Utilisateur	mettre en pause/arrêter ma séance de sport	je pourrai consulter les résultats en faisant un pause	-tester que le chronomètre s'arrête et recommence, que le listener du gps fait pareil, tester l'affichage des	3		3	2
10	Android/Ergonomie	Utilisateur	avoir une meilleure ergonomie sur mon application	Je pourrai mieux consulter les informations de ma séance de sport	-meilleur appareance et design plus ergonomique	4		3	2
11	Android/Ergonomie	Utilisateur	attendre que mon GPS se callibre	Je pourrai avoir une meilleure précision durant ma séance de sport et les données ne seront pas faussées	-test que un temps d'attente est émis lors du run, que le GPS est mieux calibré, affichage progress bar	1		5	3
12	Android/Données	Utilisateur	sauvegarder mes données de localisation	Je pourrai consulter la carte avec le trajet effectué sur mon application web avec un mappage sur l'effort	-tester que les données des listes longitude, latitute, vitesse, altitude soient bien sauvées	2		8	3
20	Site/Connexion	Utilisateur	log in sur mon site avec mon compte Google	Je pourrai voir mes résultats ainsi que les données de mes vêtements	-tester les champs de connexion apparaissent et que l'utilisateur puisse se connecter	5		3	2
21	Site/Connexion	Utilisateur	lougout du site	Je pourrai changer de compte	-tester que l'utilisateur n'est plus connecté, tester affichage page login(redirection)	5		3	3
22	Site/Données Android	Utilisateur	consulter un historique de mes séances	Je pourrai analyser mes résultats	-tester que les données des séances s'affichent correctement	3		5	3
23	Site/Données Android	Utilisateur	voir mes informations de ma dernières séance sur la première page	Je pourrai analyser mes résultats	-tester que les données s'affichent et sont bien retournées du cloud	4		5	3
24	Site/Données vêtement	Utilisateur	voir mes données des capteurs	Je pourrai analyser les résultats du vêtement	-tester que les données apparaissent et soient juste	1		8	4
25	Site/Données Android	Utilisateur	voir la carte avec mon trajet effectué et avoir l'effort mapper dessus	Je pourrai mieux ananlyse mon parcours	-tester que la carte s'affiche et se construir bien, tester si la vitesse s'inscrit et éventuellement que le trajet change de couleur	2		13	4
26	Site/Analyse Données	Utilisateur	comparer deux séances de sport	Je pourrai analyser ces données sur des graphiques	-tester que les données des capteurs depuis l'API sont bien récupérés, les afficher dans des graphiques pertinents	3		8	4
27	Site/Profile	Utilisateur	voir mes infos de profile, contenant l'age, la taille, le sexe	Je pourrai consulter et changer les valeurs	-tester que les infos apparaissent, save dans datastore les données	4		5	4

Annexe III : Sprintbacklog

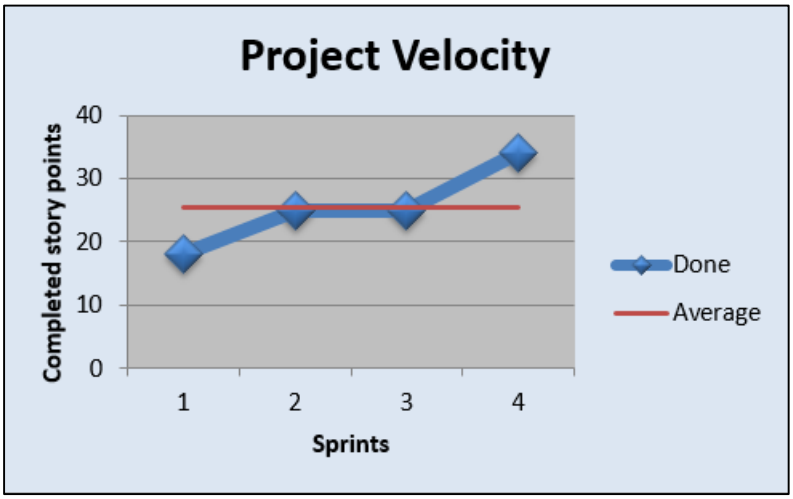
US Nr.	Task	Planification (hours)	Workload (hours)	Status	Resp.	Done	Impediments
	Sprint 1						*(14.05.2014 - 27.05.2014)*
1	Créer l'activité login et lié avec google account	2	2	DONE	Vincent	14.05.2014	
1	Lien avec les écrans	1	1	DONE	Vincent	16.05.2014	
2	Créer les écrans principaux	2	2	DONE	Vincent	16.05.2014	
2	Sauvegarder les données et les passer à la bonne activité	1	1	DONE	Vincent	16.05.2014	
3	Créer les écrans maps et nouvelle séance	2	2	DONE	Vincent	19.05.2014	
4	Intégrer Google maps et GPS	4	5	DONE	Vincent	19.05.2014	
5	Dessiner trajet sur Google maps (Polylines)	2	2	DONE	Vincent	19.05.2014	
5	Intégrer listener off après clique "Stop"	2	2	DONE	Vincent	21.05.2014	
6	Créer écran	1	1	DONE	Vincent	23.05.2014	
6	Passer les variable à l'écran résumé séance	1	1	DONE	Vincent	23.05.2014	
6	Créer les formules de calculs (Duration, calories brûlées) et les implémentés dans écrans	2	2	DONE	Vincent	26.05.2014	
1	Divers modifications	3	2	DONE	Vincent	26.05.2014	
		23	23				
	Sprint 2						*(27.05.2014 - 12.06.2014)*
7	Implémentation méthode de calcul pour distance effectuée et min par km, liste latitutes et	3	3,5	DONE	Vincent	30.05.2014	
7	Afficher les valeurs dans séance en cours et les reporter sur résumé séance pour sauvegarde futur	1	1	DONE	Vincent	30.05.2014	
8	Création méthode de test dans backend api pour tester l'ajout de données dans le datastore	2	2	DONE	Vincent	04.06.2014	
8	Appel de la méthode depuis android et test de sauvegarde de données fictives android dans cloud	2	2	DONE	Vincent	04.06.2014	
9	Ajout bouton play/pause sur écran et implémentation	4	4	DONE	Vincent	03.06.2014	
10	Amélioration ergonomie et design de l'application	2	2	DONE	Vincent	03.06.2014	
8	Création méthode pour ajout d'une nouvelle séance et auvegarde dans le cloud	3,5	3,5	DONE	Vincent	06.06.2014	
8	Passage appengine de localhost à version en ligne (avec endpoint)	1,5	1,5	DONE	Vincent	06.06.2014	
8	Ajout adresse email et date pour sauvegarde séance dans le cloud	1,5	1,5	DONE	Vincent	09.06.2014	
20	Création page index et login pour site web + appication bootstrap	2,5	2,5	DONE	Vincent	09.06.2014	
20	Intégration google charts sur site et page comparatif séance	2	1,5	DONE	Vincent	09.06.2014	
		25	25				

	Sprint 3						*(12.06.2014 - 02.07.2014)*
12	Création d'une méthode pour sauvegarder collection de (latitudes, longitudes, vitesses, altitudes)	4,5	4,5	DONE	Vincent	14.06.2014	
12	Intégration des autres méthodes pour sauvegarde des données de la map	1,5	1	DONE	Vincent	14.06.2014	
11	Intégration d'un progress bar et thread pour attendre que le GPS soit bien calibré avant de lancer la séance	4,5	4,5	DONE	Vincent	16.0.6.2014	
11	Code Review + nettooyage code + amélioration police de l'application	2	2	DONE	Vincent	17.06.2014	
21	Intégration du logout avec compte google	1,5	1,5	DONE	Vincent	23.06.2014	
22	Création page historique	0,5	0,5	DONE	Vincent	23.06.2014	
22	Création des méthodes côté api explorer	1,5	1,5	DONE	Vincent	24.06.2014	
22	Appel des méthodes du côté du site et création des classes	2	2	DONE	Vincent	25.06.2014	
22	Affichage dans une table des résultats + ajout tri sur table	2	2	DONE	Vincent	26.06.2014	
22	Divers modification design	2	2	DONE	Vincent	27.06.2014	
23	Intégration design pour première page (icon bootstrap)	2	2	DONE	Vincent	30.06.2014	
23	Appel de la dernière séance et affichage sur page index	4	4,5	DONE	Vincent	30.06.2014	
		28	28				
	Sprint 4						*(07.07.2014 - 14.07.2014)*
24	Création méthode rest pour get les donnée en	3	3	DONE	Vincent	03.07.2014	
24	Gérer le json et substring les données	4	4	DONE	Vincent	03.07.2014	
24	Affichage sur site	1	1	DONE	Vincent	03.07.2014	
25	Création de la map sur index et compare séance	3	3	DONE	Vincent	04.07.2014	
25	Appel des données de localisation par les	2	2	DONE	Vincent	04.07.2014	
25	Ajout des données dans la map en javascript et différents tests	2	2	DONE	Vincent	04.07.2014	
25	Ajout des marqueurs sur la carte pour meilleur représenation et test	3	3	DONE	Vincent	05.07.2014	
25	Améliorations rendu carte	2	2	DONE	Vincent	05.07.2014	
26	Appel des données des deux séances différentes	3	3	DONE	Vincent	06.07.2014	
26	Mise en place de la page avec les différents éléments (bouton, design, graphiques)	3	3	DONE	Vincent	10.07.2014	
26	Améliorations rendu index	4	4	DONE	Vincent	14.07.2014	
26	Amélioration rendu compare séance	5	5	DONE	Vincent	15.07.2014	
27	Création page profil	1	1	DONE	Vincent	15.07.2014	
27	Création méthod pour sauvegarde dans Datastore	1	1	DONE	Vincent	15.07.2014	
27	Appel dans android et sauvegarde	1	1	DONE	Vincent	15.07.2014	
27	Affichage sur site + sauvegarde	2	2	DONE	Vincent	15.07.2014	
		40	40				

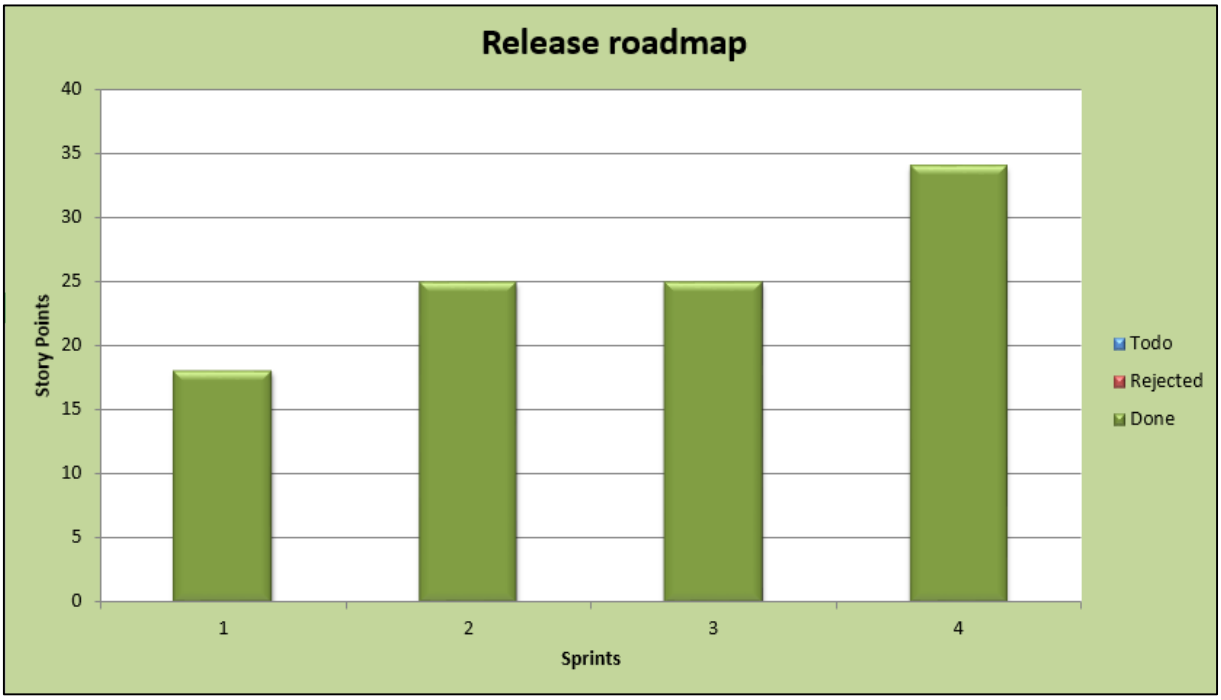
55

[55] Ici seules les heures passées pour les User Stories ont été stipulées. Pour avoir toutes les heures en détails voir dans le fichier *Heures.xlsx*

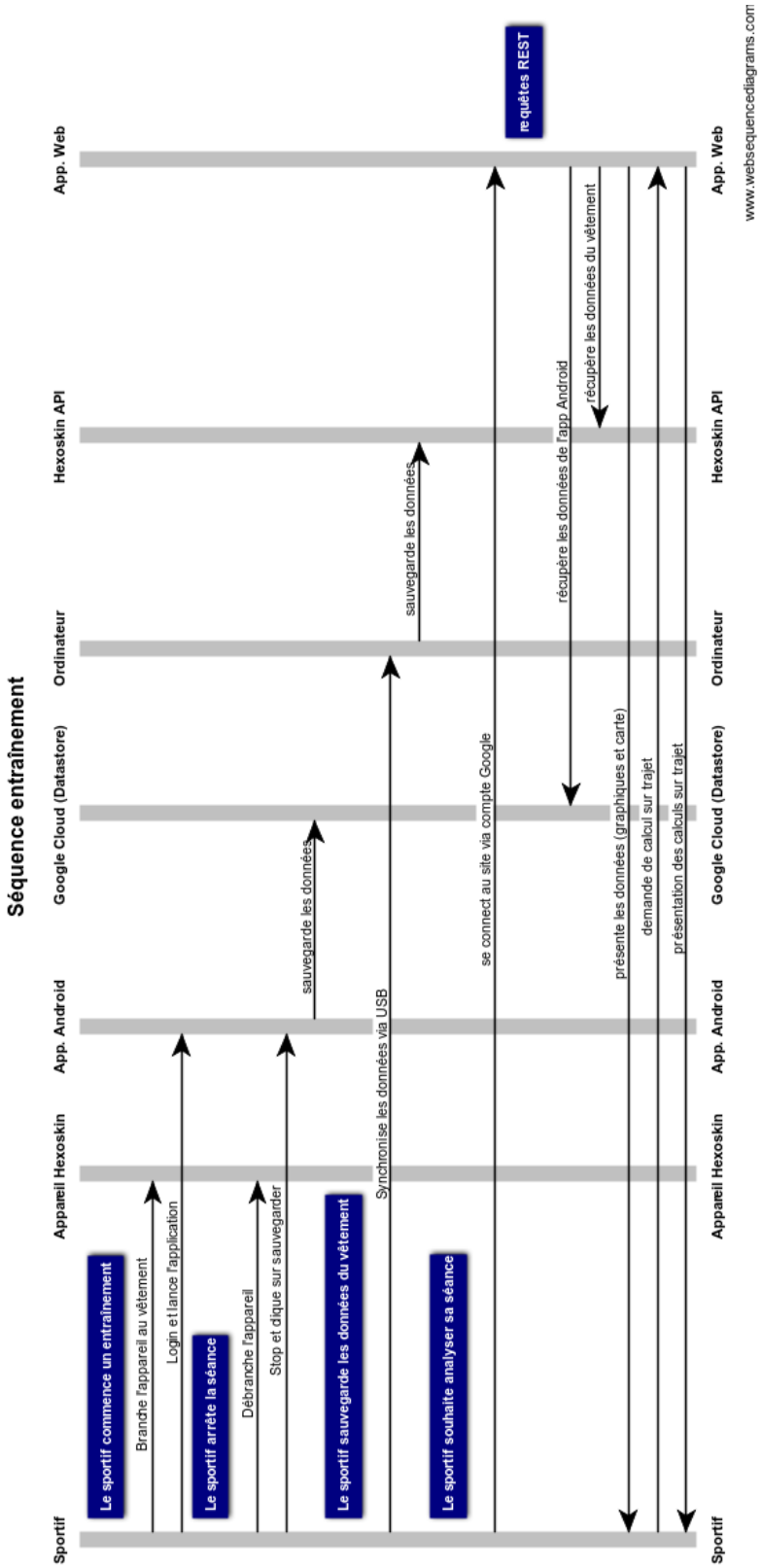
Annexe IV : Release Roadmap / Velocity



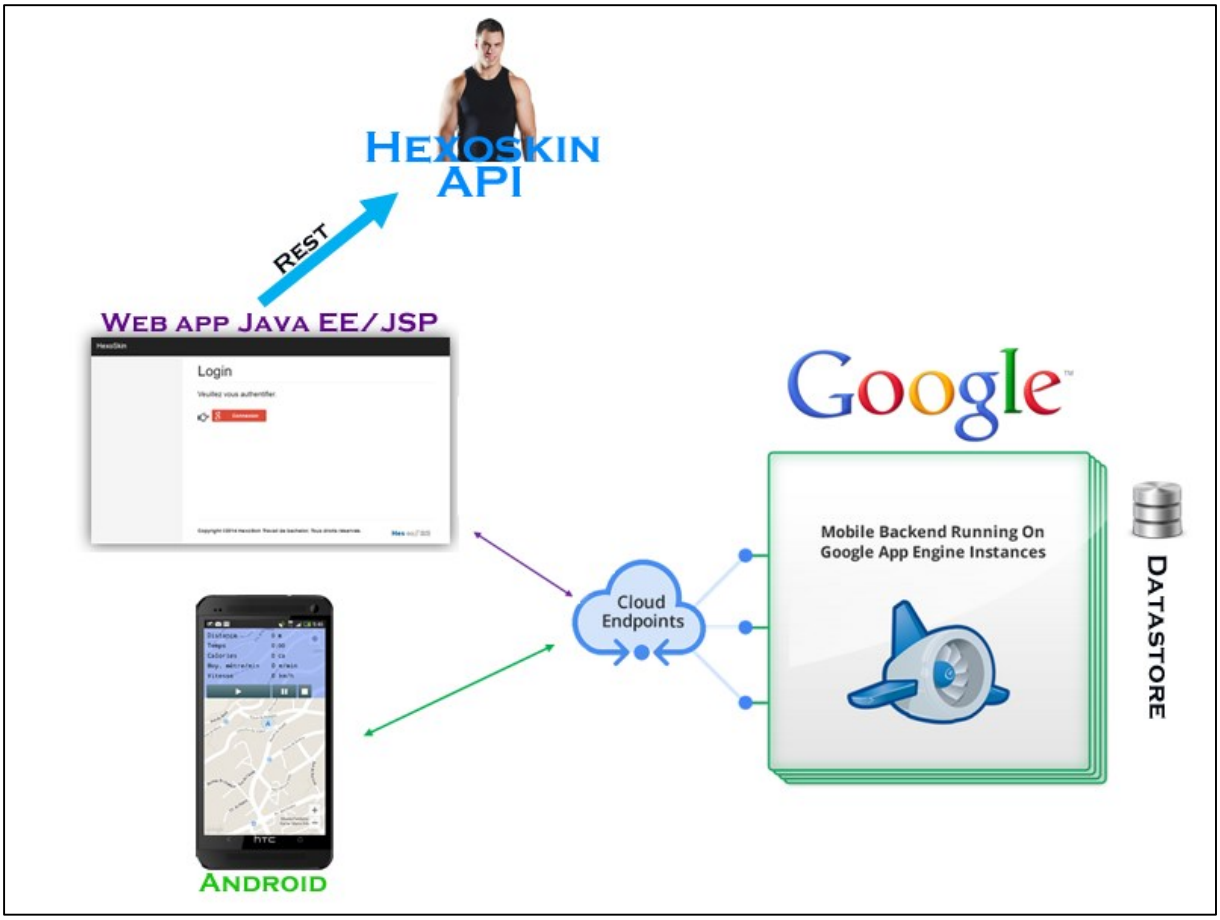
	Sprints			
	1	2	3	4
Done	18	25	25	34
Todo	0	0	0	0
Rejected	0	0	0	0
Realized	18	25	25	34
Average	25,5	25,5	25,5	25,5
Total	18	25	25	34



Annexe V : Diagramme de séquence



Annexes VI: Architecture



Déclaration sur l'honneur

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

Monsieur Michael Schumacher

Monsieur Bruno Alves

Sierre, le 28 juillet 2014

Pont Vincent